

Apéndice A

ATP-EMTP

A.1 ¿Qué es ATP?

El Programa de Transitorios Alternativo (*Alternative Transients Program*, ATP) es considerado un programa informático universal para la simulación de fenómenos electromagnéticos transitorios y de naturaleza electromecánica en sistemas eléctricos de potencia. Complejas redes eléctricas y sistemas de control pueden ser de esta manera simulados.

El Programa de Transitorios Electromagnético (*Electromagnetic Transients Program*, EMTP) fue desarrollado en el dominio público de la Administración de Energía Bonneville (*Bonneville Power Administration*, BPA) de Portland, Oregon, como paso previo a su comercialización en 1984 por el Grupo de Coordinación de Desarrollo del EMTP (*EMTP Development Coordination Group*) y el Instituto de Investigación de Energía Eléctrica (*Electric Power Research Institute*, EPRI) de Palo Alto, California. Varios expertos en todo el mundo, sin embargo, han contribuido a su posterior desarrollo y mejora en años posteriores.

Protegido por las leyes de Estados Unidos, ATP puede ser adquirido bajo licencia, expedida y garantizada por los propietarios del material.

A.2 Requerimientos de hardware y software para ATP.

ATP está disponible para la mayoría de plataformas PC basadas en Intel: DOS, Windows 3.1/9x/NT, OS/2, Linux, etc., y para otras computadoras (Digital Unix y VMS, Apple Macs, etc.)

Un Pentium PC con una configuración mínima de 128 Mb de memoria RAM, 20 Mb de espacio libre en disco duro y VGA gráficos es suficiente para ejecutar ATP bajo MS-Windows.

Las versiones más populares de compiladores son las siguientes:

- MS-Windows 9x/NT/2000/XP:

32 bit GNU-Mingw32 y Watcom ATP.

- MS-DOS, MS-Windows 3x/95/98:

32 bit Salford ATP (requiere DBOS/486).

- Linux:

Versión GNU de ATP.

A.3 Licencia de ATP.

Para bajar el programa de internet ha de obtenerse el permiso o licencia del Grupo de Usuarios de EMTP Canadá/América (*Canadian/American EMTP User Group*) o un grupo de usuarios regional autorizado. Mayor información al respecto puede encontrarse en la página web www.emtp.org.

A continuación se muestra la localización geográfica de los distintos grupos de usuarios de ATP-EMTP y de detalles de contacto:



Figura. Localización de grupos de usuarios de ATP-EMTP.

Nombre	E-mail y dirección www
Canadian/American EMTP User Group	canam@emtp.org
European EMTP-ATP Users Group (EEUG)	eeug@emtp.org , www.eeug.org
Japanese ATP User Group (JAUG)	jaug@emtp.org , www.jaug.jp/~atp/index-e.htm
Latin American EMTP User Group (CLAUE)	clau@emtp.org , www.furnas.gov.br/atp
Argentinian EMTP User Group (CAUE)	caue@emtp.org http://iitree.ing.unlp.edu.ar/estudios/caue/caue.html
Australian EMTP User Group (AEUG)	aeug@emtp.org
Korean EMTP User Group (KEUG)	keug@emtp.org
Republic of China EMTP User Group	d023@taipower.com.tw
Indian EMTP User Group (IEUG)	malathi@bom4.vsnl.net.in
South African ATP User Group (SAAUG)	j.vancoller@ee.wits.ac.za , www.ee.wits.ac.za/~atp

Tabla. Direcciones email y páginas web de grupos de usuarios de ATP-EMTP.

A.4 Capacidades del programa.

La siguiente tabla muestra los límites máximos de algunos componentes que se pueden utilizar en la simulación de un circuito de potencia:

Líneas	6000
Elementos lineales	10000
Interruptores	1200
Fuentes	900
Elementos no lineales	2250
Máquinas síncronas	90

A.5 Descripción del entorno de trabajo.

A fin de facilitar su uso, ATP-EMTP permite la integración de una serie de programas complementarios: editores de texto, un editor gráfico de circuitos eléctricos y aplicaciones específicas para la visualización gráfica de los resultados obtenidos. Se accede a dichos programas a través del gestor del entorno, ATPCC.

De esta manera, la simulación de un sistema eléctrico o electromecánico puede iniciarse con la creación del modelo en el editor gráfico (fichero de extensión **.adp** o **.cir**). La ejecución (*run ATP*) del modelo en el mismo editor genera el fichero **.atp** y los ficheros de salida **.lis** y **.p14**, éstos últimos registran los resultados de la simulación efectuada. En un editor gráfico puede visualizarse a partir del fichero de extensión **.p14** los resultados obtenidos de esta manera.

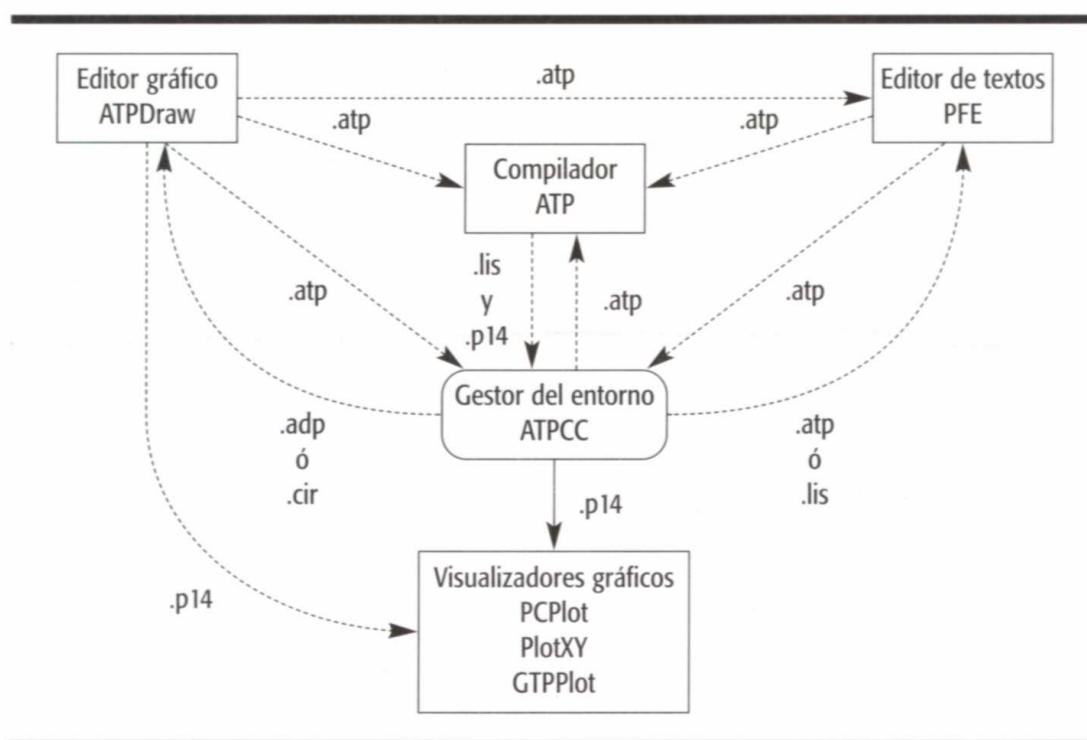


Figura. Interacción entre los programas de ATP-EMTP.

A.5.1 Gestor del entorno: ATPCC.

Como se ha mencionado, *ATP Control Center* (ATPCC) facilita el acceso al conjunto de programas que pueden formar parte del paquete ATP-EMTP (el editor gráfico ATPDraw, el editor de textos PFE32, los programas de representación de resultados PCPlot, PlotXY y GTPPlot) o ejecutar directamente el propio ATP.

Se ha de hacer notar que permite el acceso o integración de hasta diez programas adicionales, así como la posibilidad de trabajar con dos versiones de ATP.

A.5.1.1 Pantalla principal.

Al ejecutar ATPCC aparece la pantalla principal del programa. Se muestra en la figura siguiente:

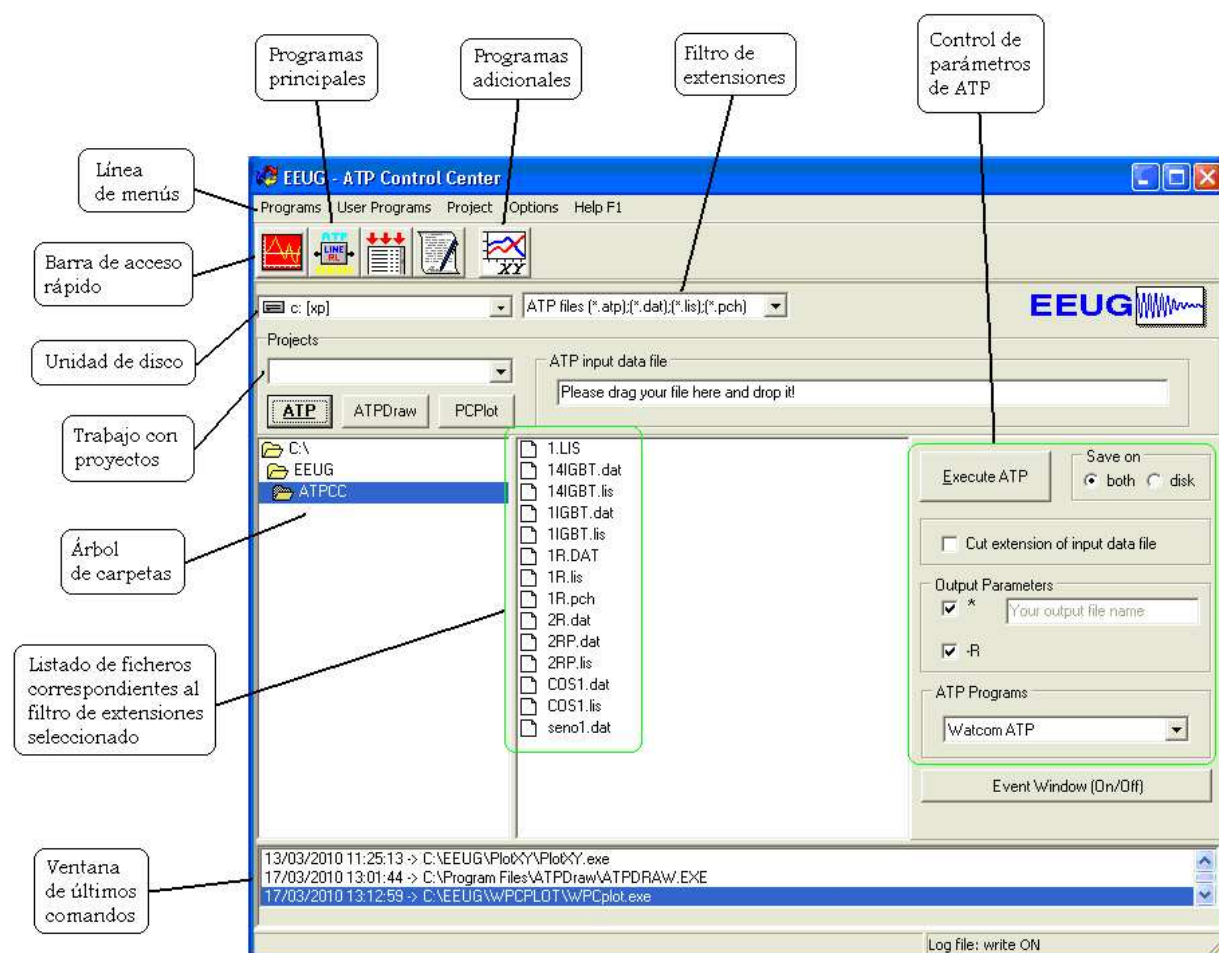


Figura. Pantalla principal del programa ATPCC.

A.5.1.2 Configuración de los programas principales.

La configuración de los programas principales (ATP, ATPDraw, PCPlot y el Editor) se realiza a través de la opción **Main Programs Settings** del submenú **Options**.

Se muestra de esta manera la siguiente ventana de diálogo, compuesta por dos pestañas. La primera de las cuales hace referencia a las dos versiones del programa ATP que posteriormente se podrán ejecutar desde ATPCC:

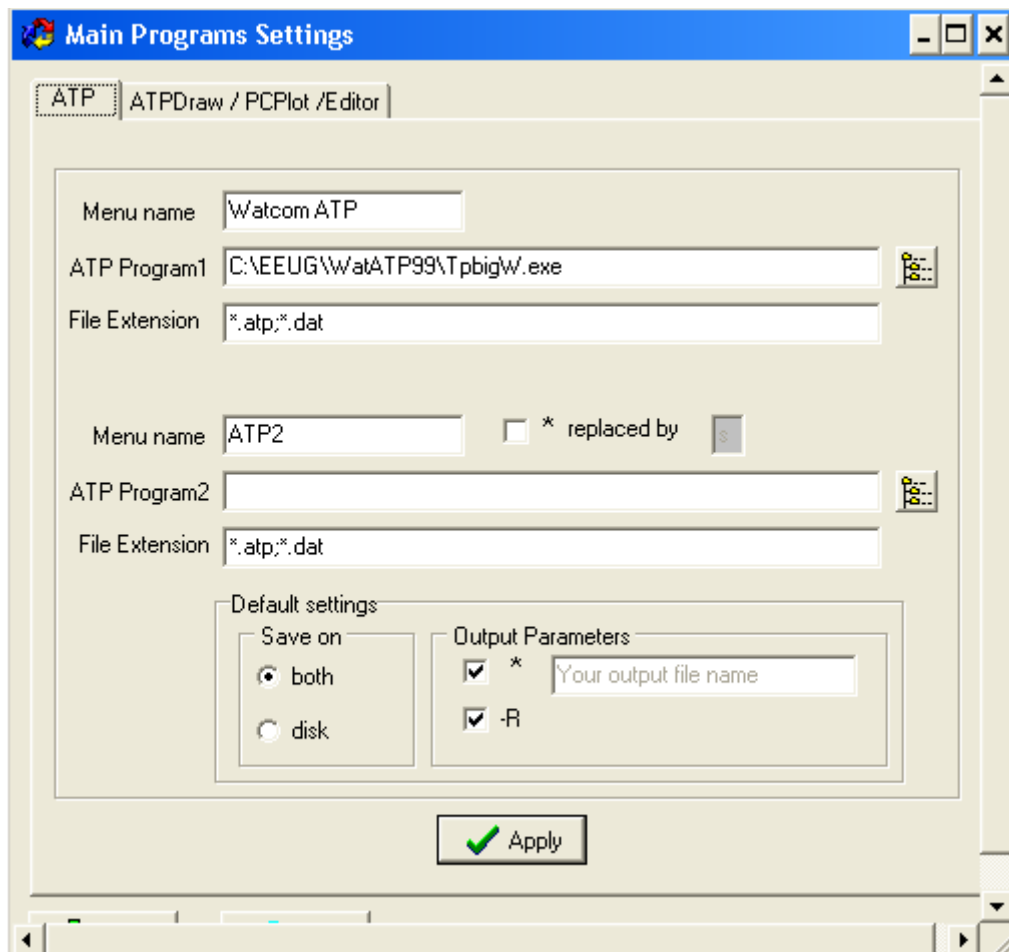


Figura. Ventana de diálogo para configuración de programas principales: ATP.

Las opciones por defecto indican los siguientes aspectos:

- **Save on:** salvar los resultados obtenidos en un fichero de extensión **.lis (disk)** o presentarlos también en la pantalla **(both)**.
- **Output Parameters:**

«*»:se marca esta casilla cuando se desea que el nombre del fichero de salida coincida con el nombre del fichero **.atp** utilizado como entrada. En caso contrario se especifica el nombre del fichero de salida (incluyendo su extensión **.lis** o **.out**).

«-R»:se selecciona esta opción si se quiere que, cuando ya exista un fichero **.lis** correspondiente al fichero **.atp** utilizado como entrada, el resultado de una nueva simulación se sobrescriba en ese mismo fichero.

En la pestaña correspondiente al resto de programas principales se actúa de un modo similar:

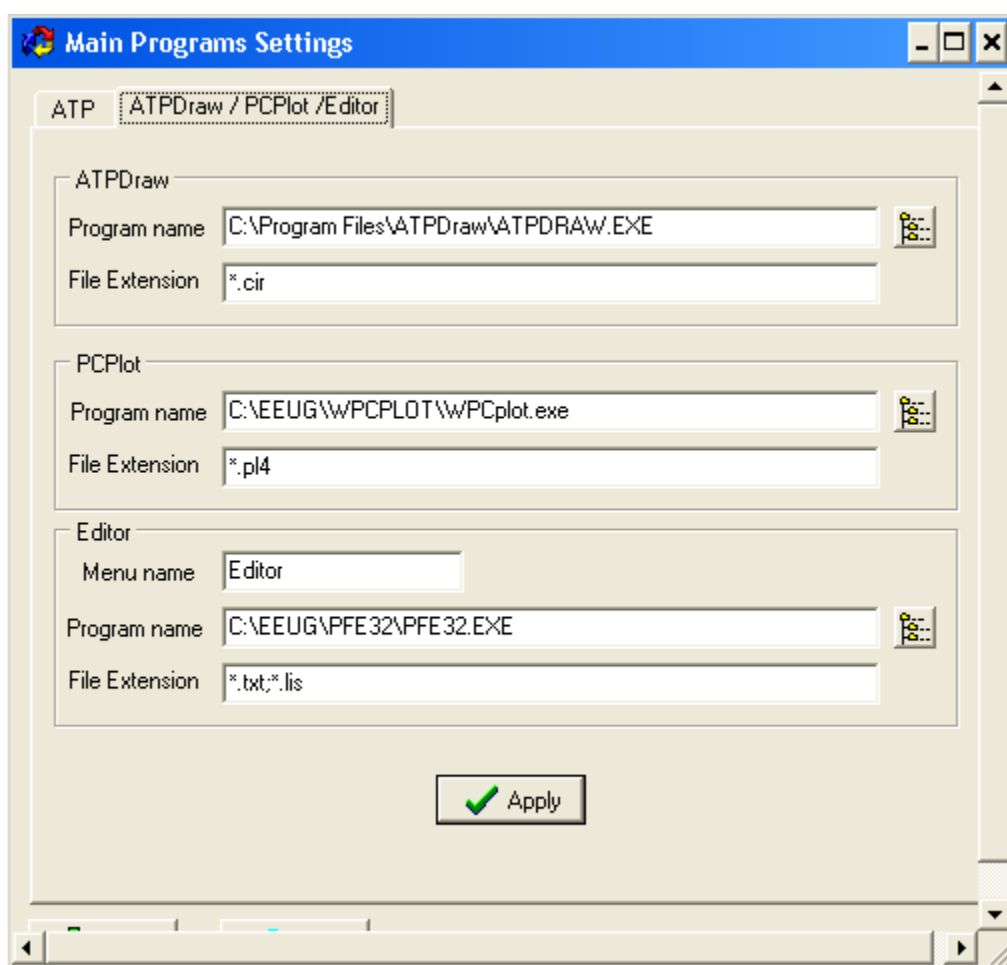


Figura. Ventana de diálogo para la configuración del resto de los programas principales.

A.5.1.3 Configuración de programas adicionales.

Para añadir, modificar o eliminar programas adicionales se selecciona, en la línea de menús, la opción **Options** → **Additional Programs**. Al hacerlo, se presenta una ventana de diálogo compuesta de tres pestañas.

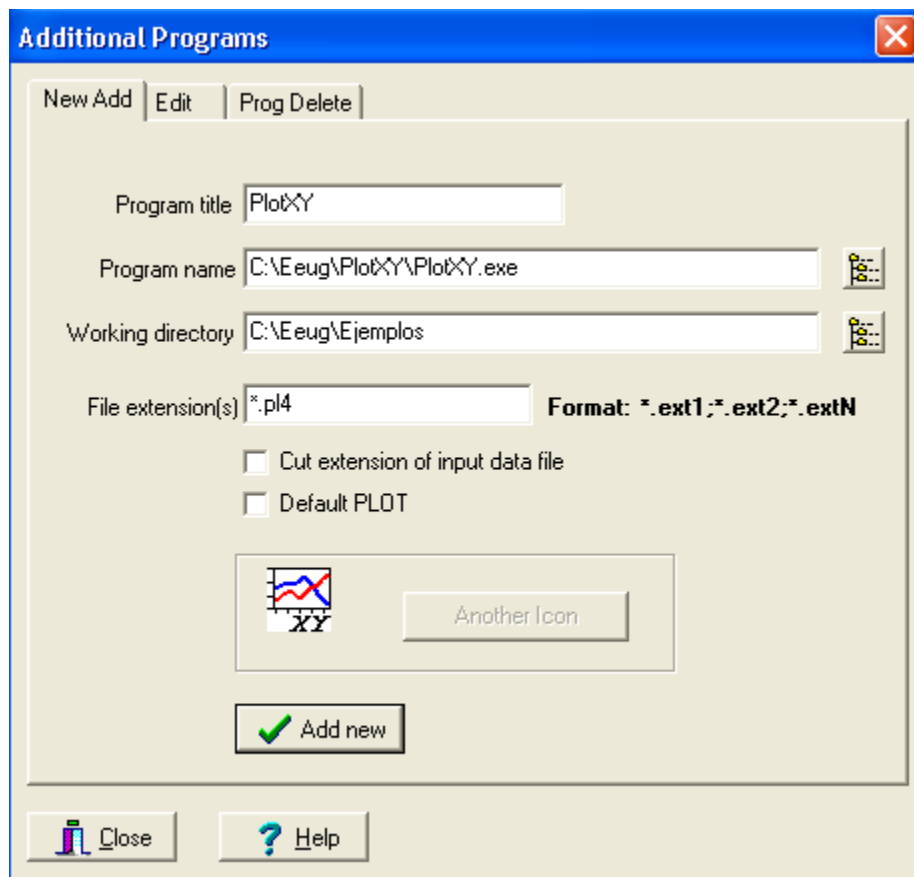


Figura. Incorporación de programas adicionales.

Las siguientes opciones adicionales equivalen a:

- **Cut extension of input file:** elimina la extensión de los ficheros de entrada, antes de ejecutarlos con el programa adicional.
- **Default PLOT:** se utiliza cuando se pretende que el programa adicional añadido (por ejemplo: GTPPLOT o PlotXY) actúe como programa gráfico por defecto. En este caso, la extensión de los ficheros de entrada debe estar definida como **.pl4**.

A.5.1.4 Configuración de parámetros generales.

Desde la opción **Settings** del submenú **Options** pueden considerarse una serie de parámetros de carácter general.

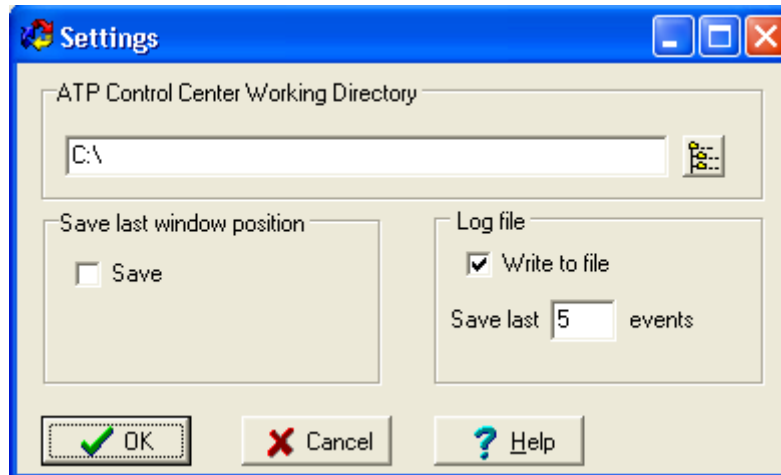


Figura. Configuración de parámetros generales.

- **ATP Control Center Working Directory:** permite definir el directorio de trabajo del programa ATPCC.
- **Log file.** Permite definir el número de eventos a recordar en la ventana de últimos comandos utilizados.
- **Save last window position.** Permite guardar la posición y el tamaño de la ventana del programa para la próxima ocasión en que se utilice.

A.6 Compilador: ATP.

Este programa constituye el núcleo fundamental de la aplicación y se trata del compilador que permite procesar los datos del circuito objeto de estudio. Maneja ficheros de tipo de texto de extensión **.dat** o **.atp**.

Las dos versiones de compiladores ATP más habituales son:

- **Watcom ATP.** Admite nombres de ficheros de gran longitud, ficheros de datos de más de 150.000 líneas y puede ejecutar simultáneamente varios casos.
- **GNU ATP.** Permite utilizar nombres largos para ficheros, proporciona un arranque rápido y requiere menos memoria.

En el esquema de la siguiente figura se muestran los módulos de simulación disponibles en ATP, las subrutinas de apoyo y la interacción entre ellos. Con todos estos elementos se pueden crear modelos de menor a mayor complejidad, que representan los diferentes equipos y componentes encontrados en los sistemas eléctricos.

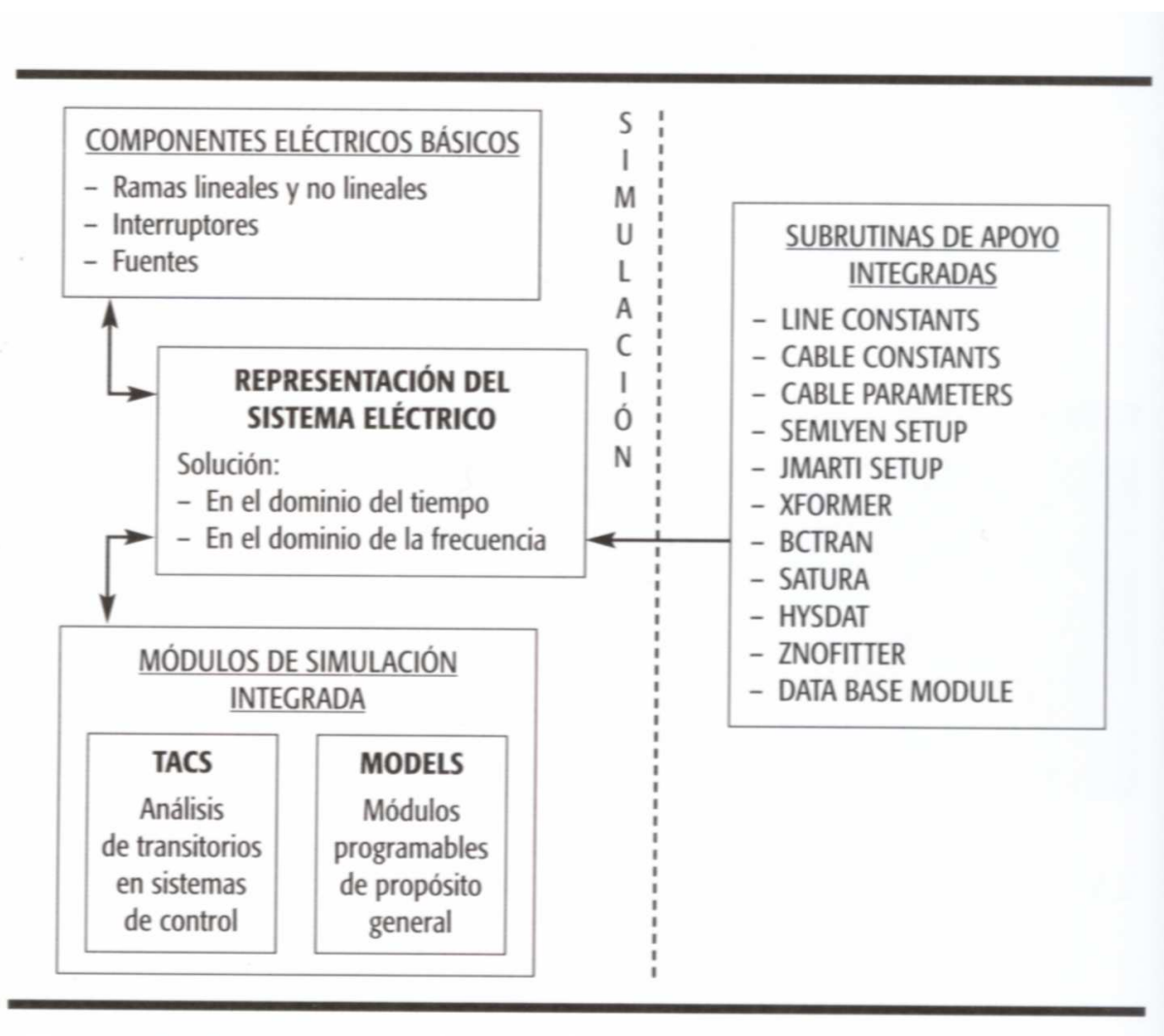


Figura. Componentes de ATP.

A.7 Editor de textos: PFE32.

El desarrollo del procesador gráfico ATPDraw ha simplificado enormemente la creación de los ficheros que son compilados posteriormente con ATP. Sin embargo, para los usuarios expertos sigue siendo necesario el empleo de un editor de textos que permita trabajar, entre otros, con los ficheros de entrada a ATP (.atp o .dat) o de salida del mismo (.lis).

Conviene señalar que la estructura de los ficheros .atp es muy rígida, ya que toda la información del sistema eléctrico a simular debe ocupar una posición específica en cada fila del fichero. Para facilitar la escritura de los datos en el fichero se incluye una serie de plantillas (**template**). Además, resulta indispensable tener a mano el manual de usuario **ATP-EMTP Rule Book**.

A.7.1 Estructura general de un fichero .atp o .dat.

El fichero comienza con una primera línea denominada BEGIN NEW DATA CASE. A continuación le siguen dos líneas MISCELLANEUS DATA CARDS, donde se permite especificar el paso de integración, tiempo máximo de la simulación, cada cuánto tiempo almacena puntos para su posterior representación, etc.

Tal y como se ha comentado, puede adjuntarse una plantilla al editor de textos PFE32. Ello puede hacerse mediante la opción **Template** → **Attach File**. Una vez adjuntada, se puede utilizar sin más insertándola en el fichero sobre el que se está trabajando, con la opción **Template** → **Edit**.

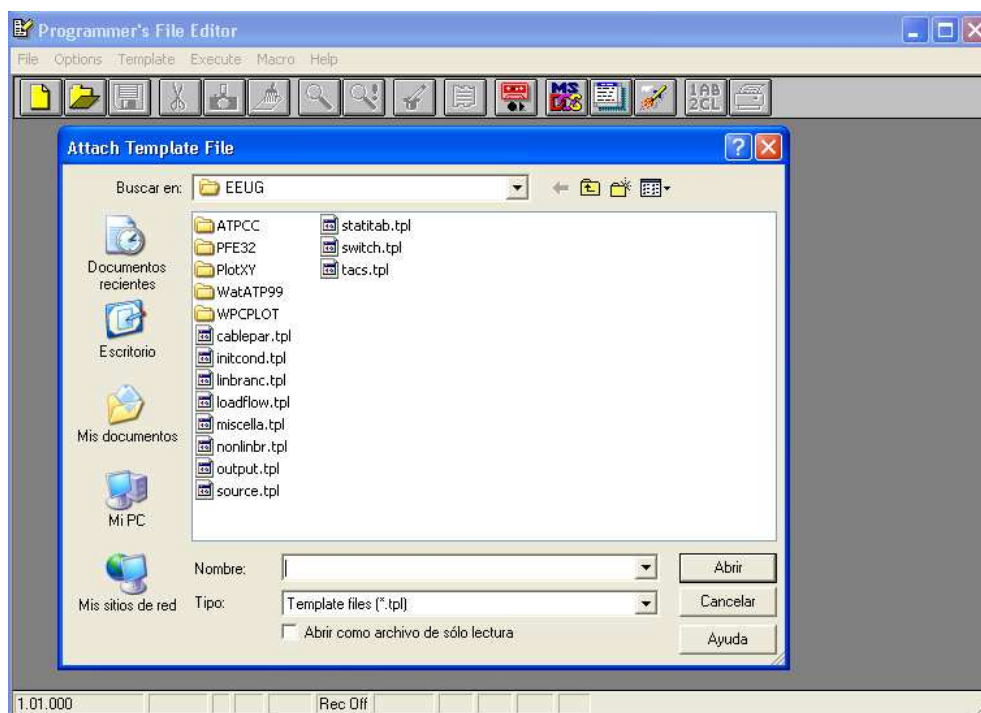


Figura. Inserción de plantillas.

Precediendo a las líneas MISCELLANEUS pueden utilizarse una serie de órdenes que permiten la realización de cálculos especiales, como, por ejemplo: flujos de carga, cálculo de armónicos, etc. Éstas pueden ser:

- CABLE CONSTANT
- DATA BASE MODULE
- FREQUENCY SCAN
- HARMONIC FREQUENCY SCAN
- HYSTERESIS
- LINE CONSTANT

- SATURATION
- XFORMER
- ZNO FITTER
- FIX SOURCE (esta opción permite calcular un flujo de cargas)

Una fila de comentarios empieza con la letra 'C'.

Por otra parte, en cualquier línea del programa pueden especificarse las siguientes tarjetas especiales:

\$OPEN	\$CLOSE	\$SWIDTH	\$DISABLE
\$ENABLE	\$NEW EPSLIN, EPS	\$MONITOR	\$LISTOFF
\$LISTON	\$VINATGE, M	\$STOP	\$WATCH
\$COMMENT	\$DEBUG	\$UNITS	\$UNITS
\$PUNCH	\$ERASE	\$SETUP, file name	\$INCLUDE

La estructura general de un fichero .dat (o de un fichero .atp) es la siguiente:

```

BEGIN NEW DATA CASE
Miscellaneous Data Cards (2 líneas)
/TACS (si existen)                /MODELS (si existen)
(Sistemas de control)
BLANK TACS                        ENDMODELS
/BRANCH
(R, L, C, Z en general, transformadores, líneas)
/SWITCH
(Interruptores)
/SOURCE
(Fuentes de V/I ideales: rampa, escalón, sinusoidal,
máquina síncrona, motores)
/OUTPUT
(Variables de salida a representar y que se incluirán
en el fichero .pl4)
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK OUTPUT
BEGIN NEW DATA CASE
BLANK

```

A continuación se muestra un ejemplo, el cual ha sido generado por el editor gráfico ATPDraw del circuito eléctrico que se muestra a continuación:

```

BEGIN NEW DATA CASE
C -----
C Generated by ATPDRAW marzo, jueves 11, 2010
C A Bonneville Power Administration program
C Programmed by H. K. Høidalen at SEFAS - NORWAY 1994-98
C -----
C Example 1
C Your first circuit
C Rectifier bridge
C Miscellaneous Data Card ....
C dT >< Tmax >< Xopt >< Copt >
  5.E-5 .05
    500 1 1 1 1 1 0 0 1 0
C 1 2 3 4 5 6 7 8
C 3456789012345678901234567890123456789012345678901234567890

/BRANCH
C < n 1>< n 2><ref1><ref2>< R >< L >< C >
C < n 1>< n 2><ref1><ref2>< R >< A >< B ><Leng><><>0
  VA XX0002 33. 1. 0
  XX0002 33. 1. 0
  XX0021VA 33. 1. 0
  XX0021 33. 1. 0
  XX0002POS .01 1
  POS XX0021 1.E3 3
  XX0021POS 20. 3
  VS XX0025 1. 0
  VS XX0025 300. 0

/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde >< Ie ><Vf/CL0P >< type >
11VA XX0002 0
11 XX0002 0
11XX0021VA 0
11XX0021 0
XX0025VA MEASURING 1

/SOURCE
C < n 1><>< Ampl. >< Freq. ><Phase/T0>< A1 >< T1 >< TSTART >< TSTOP >
14VS 0 167.7 60. -90. A1 T1 TSTART TSTOP
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
  2POS 7.500000E+0001
  2XX0021-7.500000E+0001
  3POS XX0021 1.500000E+0002
  VS
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK

```

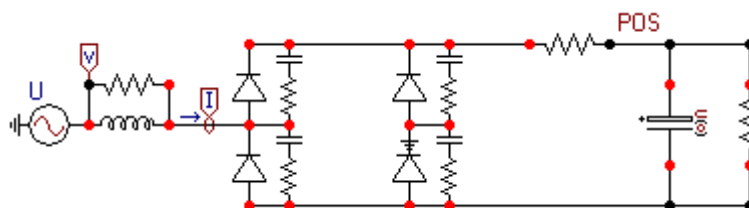


Figura. Circuito eléctrico.

A.7.2 Subrutinas de apoyo.

De forma similar, la estructura típica para las subrutinas de apoyo es la siguiente:

```
BEGIN NEW DATA CASE
Palabra especial que define la subrutina de apoyo
Datos específicos del equipo
$PUNCH
BLANK CARD
BEGIN NEW DATA CASE
BLANK
```

Ejemplo de palabras especiales son: XFORMER, BCTRAN, SATURA, HYSDAT, LINE CONSTANTS, CABLE CONSTANTS, DATA BASE MODULE, etc.

A continuación se muestra un ejemplo de DATA MODULARIZATION. El elemento creado será una resistencia.

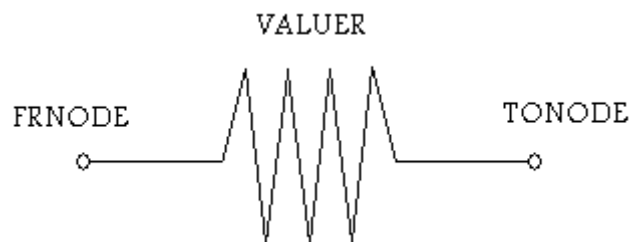


Figura. Esquema de una resistencia.

Fichero **1R.DAT**:

```
BEGIN NEW DATA CASE      --NOSORT--
C Module for a resistor
C USAGE: INCLUDE, 1R, FRNODE, TONODE, VALUER, H
DATA BASE MODULE
$ERASE
ARG, FRNODE, TONODE,
ARG, VALUER, H
NUM, VALUER, H
DUM,
/BRANCH
C<Bus  M<Bus  k<Bus  3<Bus  4<---- R<---- L<---- C<----->0
  FRNODETONODE      VALUER      H
C . . ^ . . ^ . . ^ . . ^ . . ^XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX^
```

```
BEGIN NEW DATA CASE
C Just one comment card only, since the KASEND in 'startup' file is 1.
$PUNCH
BEGIN NEW DATA CASE
BLANK
```

El comando \$ERASE clarifica el buffer donde se guardará el fichero **.pch** obtenido. DUM denota el nombre de los nodos internos del componente (en este caso no hay).

La compilación del fichero anterior genera el siguiente fichero **1R.PCH**:

```
KARD   3      3      3      3
KARG   1      2      3      4
KBEG   3      9     27     80
KEND   8     14     32     80
KTEX   1      1      0      0
$ERASE
/BRANCH
C<Bus  M<Bus  k<Bus  3<Bus  4<----R<---- L<---- C<----->0
  FRNODETONODE          VALUER          H
C . . ^ . . ^ . . ^ . . ^ . . ^xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx^

$EOF User-supplied header cards follow.                27-Aug-97  14.55.19
ARG,FRNODE,TONODE,
ARG,VALUER,H
NUM,VALUER,H
DUM,
```

El fichero 'punch' contiene los punteros numéricos de todos los argumentos usados en la plantilla. Igualmente puede obtenerse un fichero de extensión **.lib** si se escribe en el fichero **.dat** \$PUNCH name.lib.

El elemento creado, en este caso una resistencia, puede ser utilizado en la simulación de un circuito eléctrico (.dat) haciendo uso de la orden \$INCLUDE. En este caso sería:

```
$INCLUDE, 1R,FRNODE,TONODE,VALUER,H
```

FRNODE, TONODE y VALUER pueden tener un máximo de 6 caracteres.

H determina el tipo de salida: 0 para no hay salida, 1 para la corriente que circula por el elemento, 2 para la diferencia de potencial entre los terminales del elemento, 3 para disponer de ambas magnitudes (intensidad y tensión) y 4 para el consumo de potencia y energía.

A.7.3 Módulos de simulación integrada.

A.7.3.1 TACS.

El módulo TACS (*Transient Analysis of Control Systems*) se puede usar para simular el control de convertidores HVDC, sistemas de excitación de máquinas síncronas, funciones de limitación de intensidad de pararrayos, cebado de arcos en interruptores y, en general, aquellos dispositivos o fenómenos que no se pueden modelizar con los componentes eléctricos existentes en ATP.

La programación con TACS se emplea para resolver las ecuaciones diferenciales y algebraicas asociadas. Para ello, se utiliza en ATPDraw una representación basada en diagramas de bloques.

Adicionalmente, cualquier señal obtenida con el módulo TACS se puede utilizar dentro de la red eléctrica simulada, incorporándola como una fuente de tensión o intensidad, o como una señal que controle la interruptor, válvula o diodo.

A.7.3.2 MODELS.

Es un lenguaje de programación que se usa en ATP-EMTP para simular variables dependientes del tiempo con características especiales. Los ficheros MODELS se crean de forma independiente al programa principal y se pueden llamar desde cualquier programa cuantas veces se desee. Su estructura es muy similar a la de otros lenguajes de programación, como por ejemplo FORTRAN, con sentencias del tipo FOR, IF, WHILE, etcétera. Han de especificarse las siguientes declaraciones:

MODEL DEFAULT

INPUT -- Name of input variables. Variable names separated by ',' or CR

OUTPUT -- Name of output variables

DATA -- Name of data variables

VAR -- Name of local+output variables

HISTORY -- Default values of variables and expressions {DFLT:n}

INIT -- Initialization

ENDINIT

EXEC -- Execution

ENDEXEC

ENDMODEL

Cada uso de un fichero MODEL se introduce con el comando USE, especificando las entradas y salidas correspondientes a esa llamada. Los nombres de los elementos definidos en un fichero MODEL no son visibles desde el exterior y se pueden elegir sin tener en cuenta las posibles coincidencias con nombres usados fuera de este fichero MODEL.

A.8 EDITOR GRÁFICO: ATPDraw.

ATP-EMTP incluye una aplicación gráfica, denominada ATPDraw, capaz de crear y editar sistemas eléctricos para poder simularlos de forma interactiva.

La opción **Edit Commands...** (ver figura a continuación) permite el uso y ejecución de otros programas de ATP en el entorno gráfico de ATPDraw.

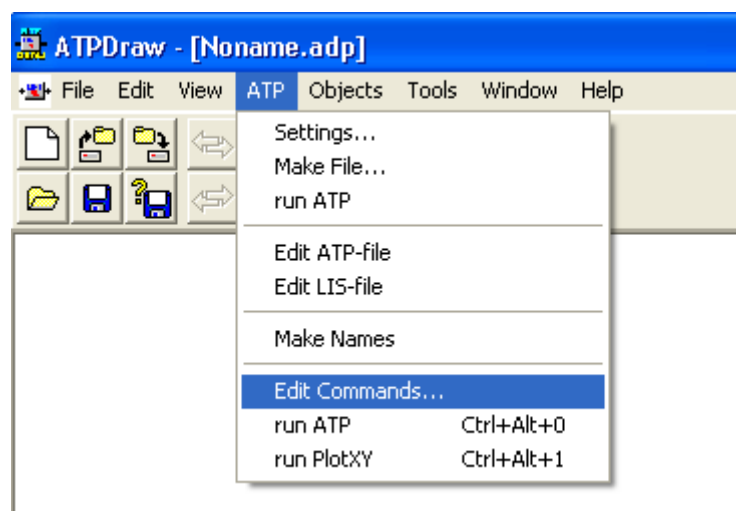


Figura. Edit Commands.

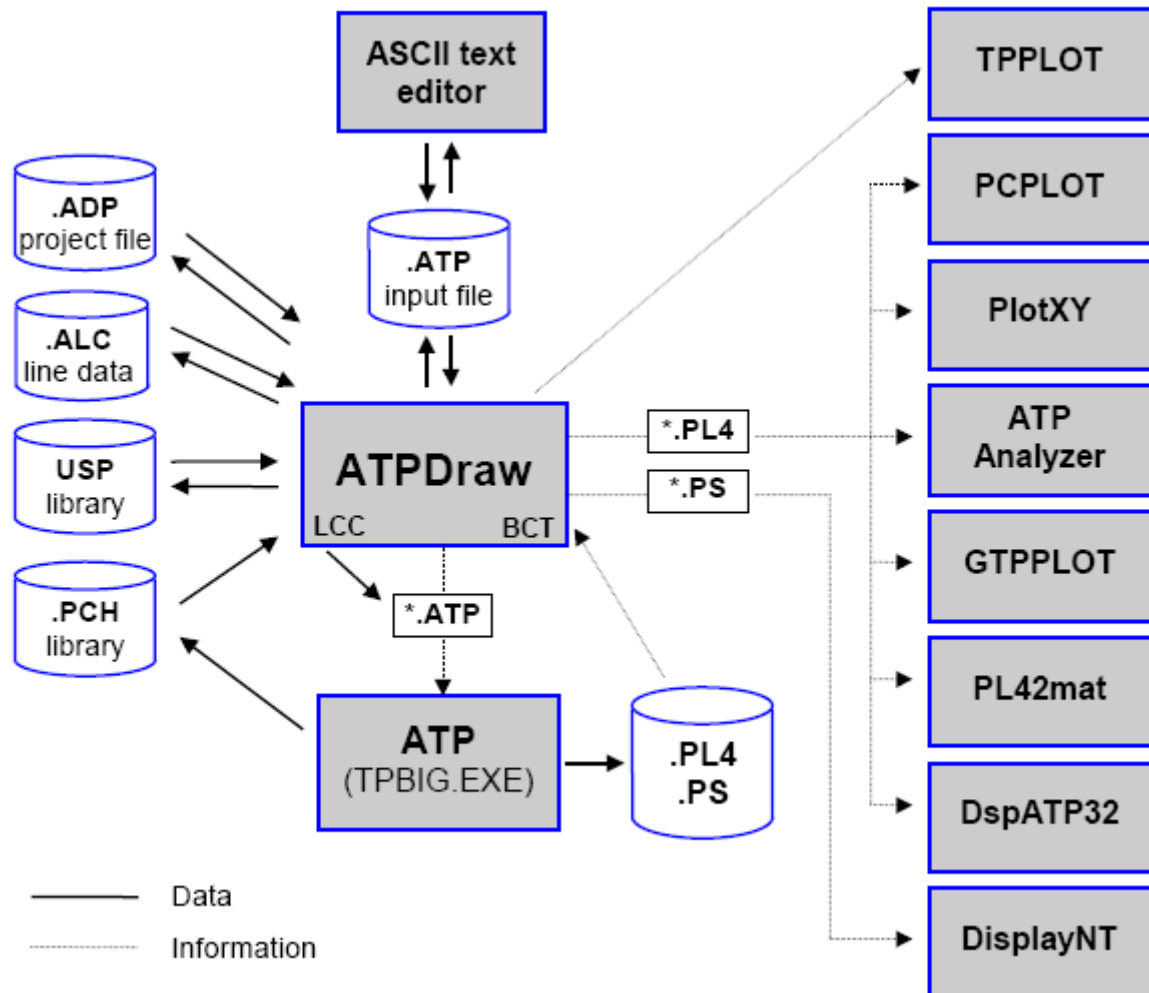


Figura. Interacción entre ATPDraw y otros programas de ATP.

A.8.1 ENTORNO DE TRABAJO.

La pantalla principal de este programa ofrece el aspecto habitual de las aplicaciones desarrolladas en entorno Windows.

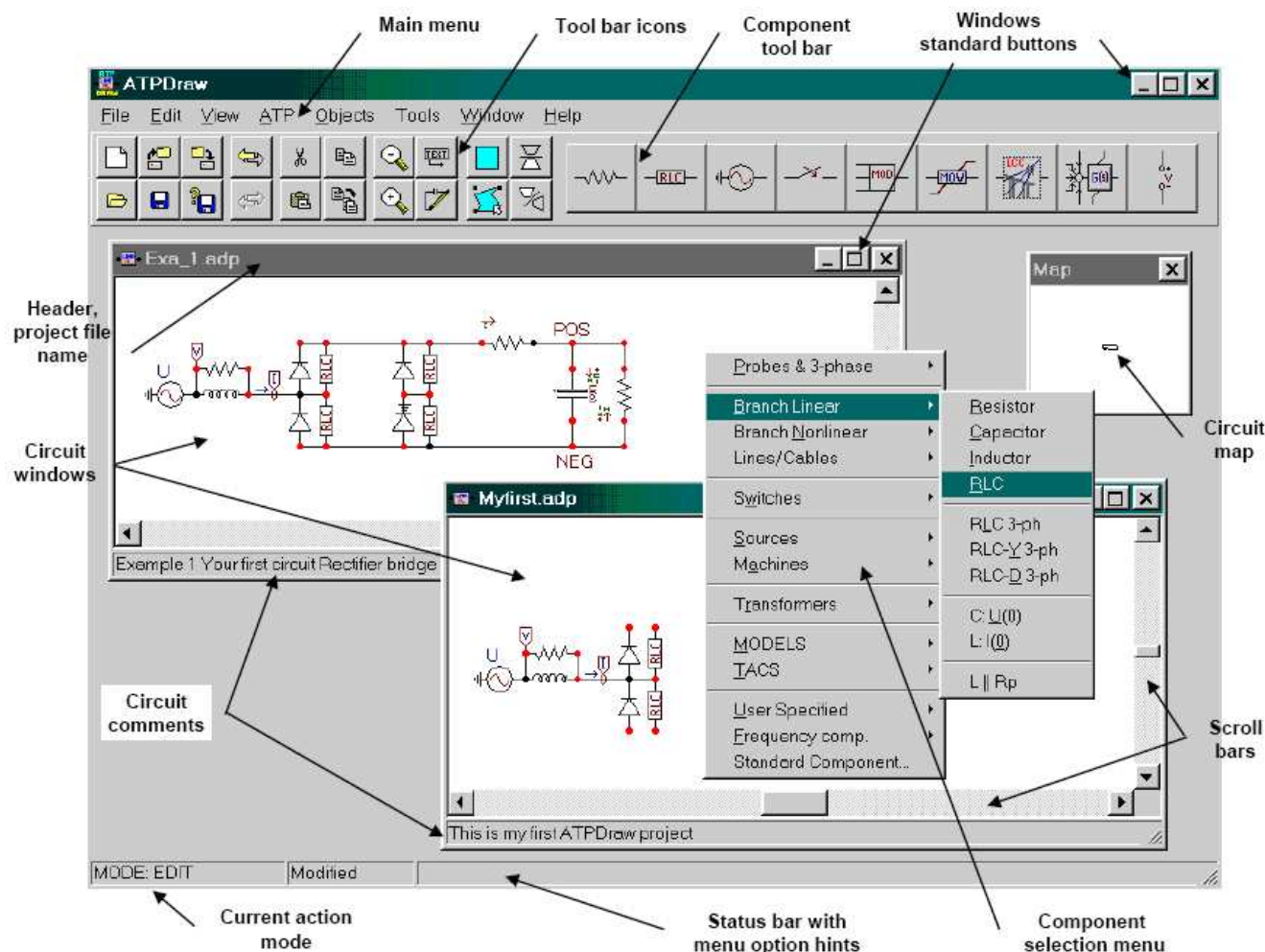


Figura. Pantalla principal de ATPDraw con el menu de selección de componentes.

A.8.3 CREAR UN CIRCUITO NUEVO.

Seleccionando el comando **New** en el menú de ficheros (*File menu*) o pinchando en el símbolo de página en blanco de la barra de herramientas, aparece una nueva ventana para la construcción de un circuito nuevo.

Una vez construido el circuito, el paso final es dar nombre a los nodos del mismo. ATPDraw puede automáticamente dar nombre a todos los nodos, sin embargo el usuario puede asimismo hacerlo, **ATP** → **Make Names**, o dar nombre solo a los nodos de especial interés. Esto último se realiza simplemente haciendo click con el botón derecho del ratón en el nodo a nombrar.

Antes de crear el fichero de entrada ATP o ejecutar la simulación, se han de especificar los parámetros de la simulación (*parameters miscellaneous*), que se encuentran en el menú **ATP** → **Settings...**

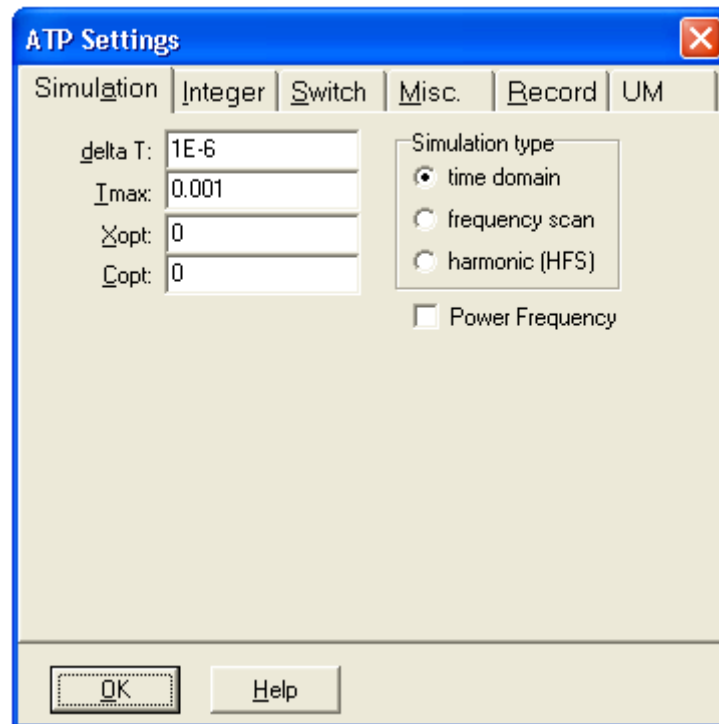


Figura. Opciones de simulación.

Donde:

- *Delta T*: Paso de tiempo en segundos.
- *Tmax*: Tiempo máximo de simulación.
- *Xopt=0*: Incuctancias en mH.
- *Copt=0*: Condensadores en uF.

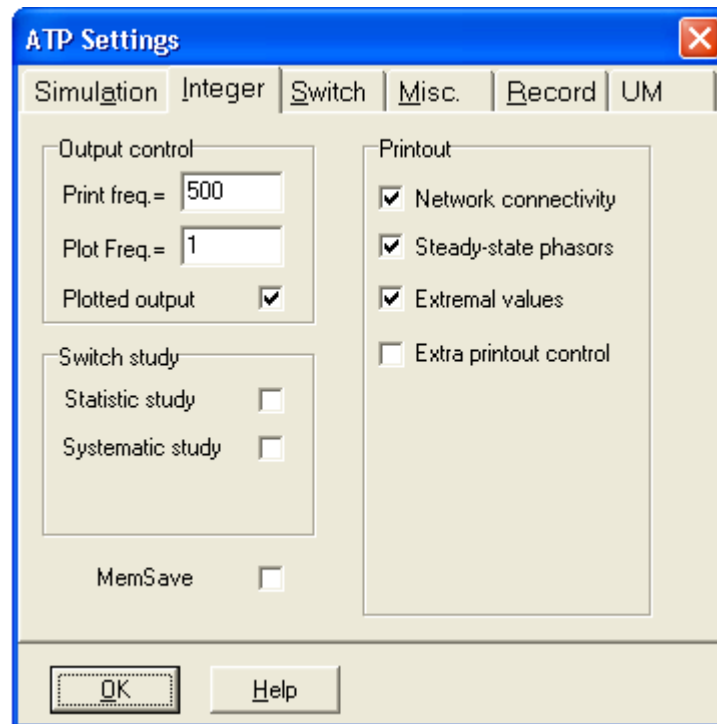
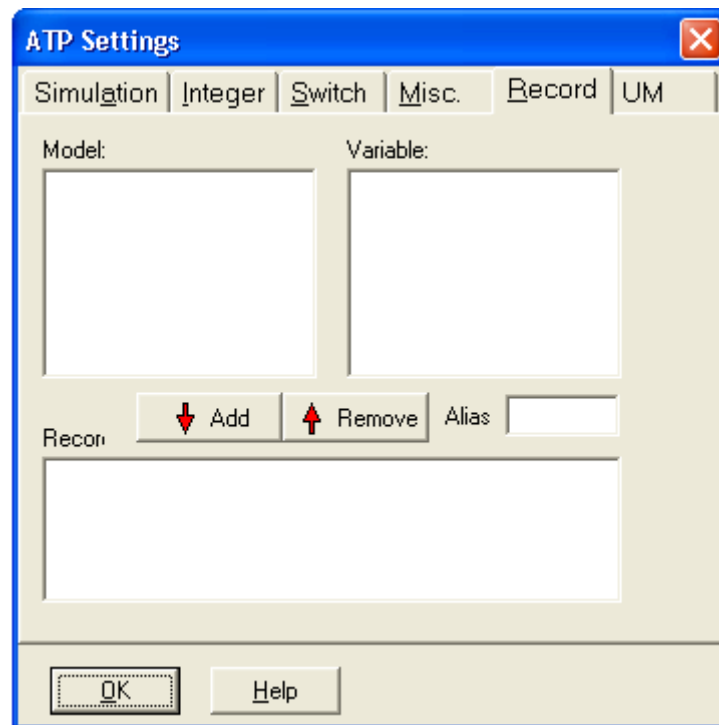


Figura. Opciones de salida.

En la figura anterior se muestra el contenido de la pestaña Integer, donde:

- **Print freq:** un valor de 500 significa que cada 500 pasos de simulación, se escribirá en el fichero **.lis**.
- **Plot freq:** un valor de 5 significa que cada 5 pasos de simulación, se escribirá en el fichero **.pl4**.

Las variables declaradas en el lenguaje de programación MODELS en la sección VAR aparecen en la pestaña **Record** en la sección *Variable*. Para grabar una variable (su evolución podrá observarse en el visualizador gráfico PCPlot) ha de seleccionarse ésta y presionar el botón *Add*. Igualmente se procede para su eliminación, presionando en este caso el botón *Remove*.


Figura. Pestaña *Record*.

La opción **ATP → Make File...** del menú principal genera un fichero ATP del circuito. Podrá guardarse en el directorio que deseemos. Finalmente se procede a su ejecución mediante la opción **run ATP** del menú **ATP**.

A.8.4 COMPONENTES DEFINIDOS POR EL USUARIO.

Además de todos los componentes estándar que contiene el editor gráfico, el usuario puede definir sus propios modelos mediante ficheros de texto. La estructura de estos ficheros puede ser mediante **MODELS** o **DATA BASE MODULE**. Estos modelos definidos por el usuario se guardan, por defecto, en los subdirectorios *Mod* y *Usp*.

El primer paso para la creación de un nuevo modelo es la generación del fichero de texto donde se describe el nuevo elemento que se quiere modelizar. Este fichero de texto se puede desarrollar mediante una de las siguientes opciones:

- Utilizando las plantillas que incluye el editor de textos PFE32.
- Escribiendo directamente desde el editor de texto, con la ayuda del ATP-EMTP Rule Book para definir correctamente la estructura del fichero.
- Utilizando la opción Data Base Module (DBM), se quiere que tenga parámetros variables.

El fichero de texto creado se tiene que compilar mediante alguna de las versiones de ATP, de modo que se obtenga otro fichero con la extensión **.lib**. Este último fichero es el que se almacena en el directorio *Usp* de ATPDraw.

Por otro lado, también hay que definir un icono y una ventana que representen el nuevo modelo de componente creado mediante el fichero **.lib**. Para ello, con la opción **Objects** → **User Specified** → **New sup-file** se crea un nuevo componente, cuya extensión es **.sup**.

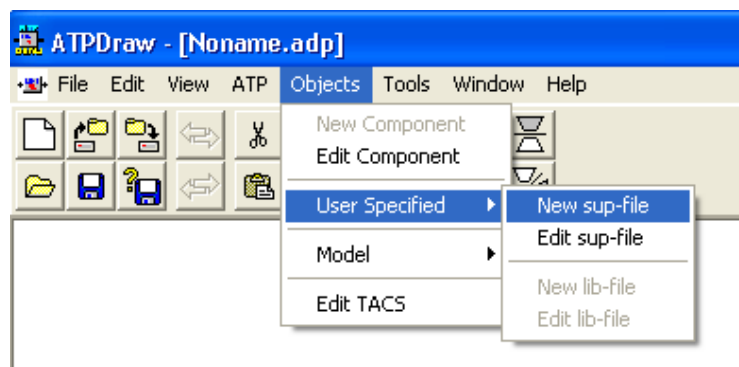
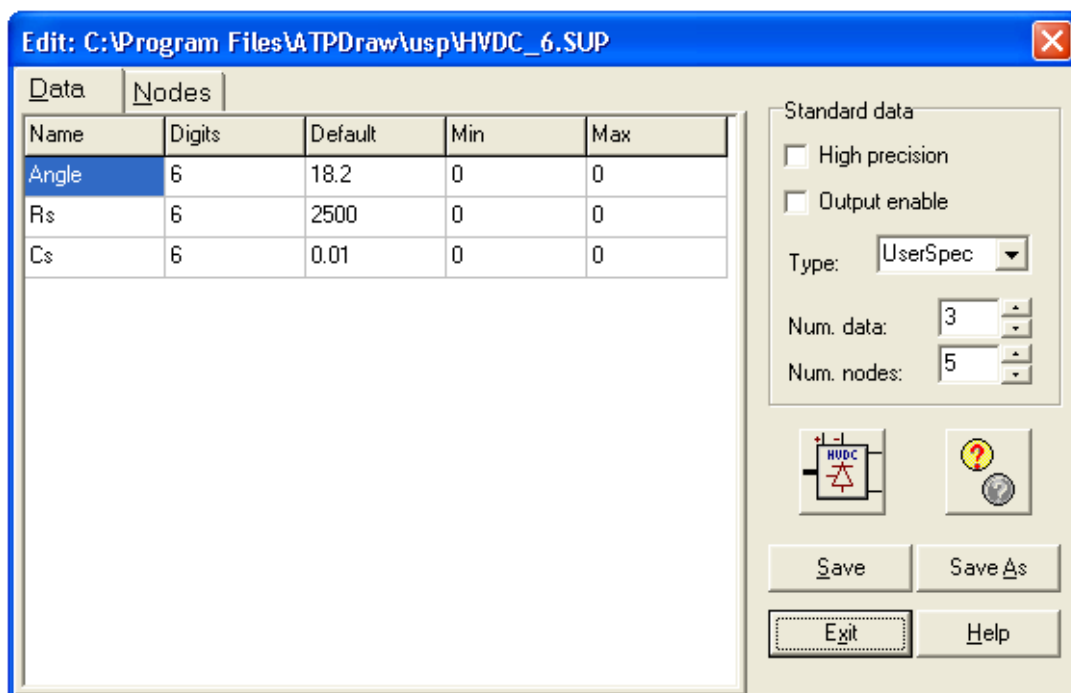


Figura. Opción New sup-file.

En la ventana que aparece a continuación hay que indicar los datos que definen dicho componente y el número de nudos disponibles para su conexión externa.



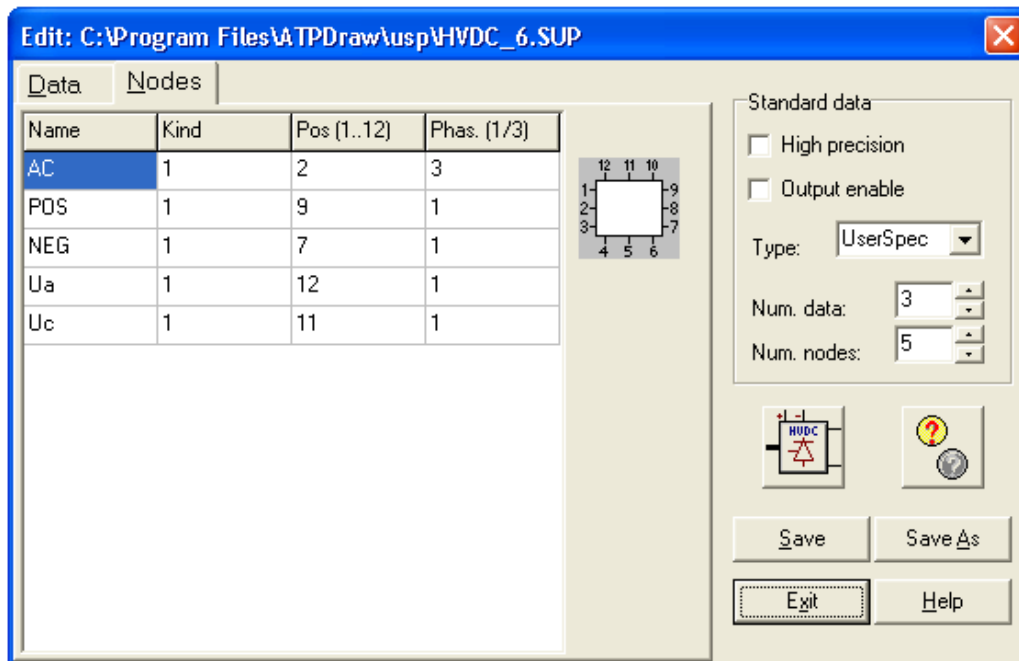


Figura. Definición de los datos y nudos del nuevo modelo.

El número de nodos y datos debe coincidir con las declaraciones ARG y NUM del fichero escrito mediante Data Base Module. Permite la introducción de hasta 36 datos, siendo el número máximo de nodos 12.

Kind:

Nodo MODELS:

- 0: nodo de salida.
- 1: nodo de entrada de intensidad.
- 2: nodo de entrada de tensión.
- 3: nodo de entrada de estado de un interruptor.
- 4: nodo de entrada de una variable de una máquina.
- 5: variable TACS.
- 6: nodo de tensión de la parte imaginaria de estado-fijo (imssv).
- 7: nodo de intensidad de la parte imaginaria de estado-fijo (imssi).
- 8: salida de otro model.

Nodo TACS:

- 0: nodo de salida.
- 1: nodo de entrada de suma positiva.
- 2: nodo de entrada de suma negativa.
- 3: nodo de entrada desconectado.

En la ventana aparecen dos botones. Pulsando el botón donde aparece una lámpara eléctrica, se abre un editor de iconos que permite diseñar el icono con el que representar el nuevo componente creado. Mediante el botón en el que aparece un interrogante, se accede a un editor de ayuda donde se puede escribir el texto que se desea mostrar, cuando se solicite ayuda desde la ventana de diálogo correspondiente a este componente.

Tras definir todos los datos, se pulsa la opción save as y se guarda el fichero con extensión .sup en el subdirectorio *Usp*, con lo que queda definido el nuevo componente. Para poder utilizar este nuevo modelo como un componente más del circuito eléctrico a diseñar, se hace uso de la opción **User specified** → **Files** y se selecciona el fichero .sup correspondiente al componente.

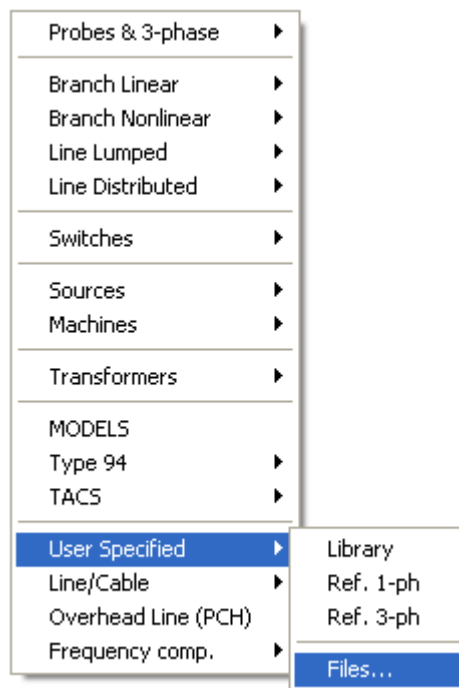


Figura. Inserción de un elemento del usuario.

Para la creación de un modelo mediante el lenguaje de programación MODELS se accede a la opción **New mod-file** del menú **Objects** (ver figura a continuación), donde se muestra la estructura a seguir (presentada en un apartado anterior) para la implementación del modelo. El manual MODELS IN ATP-Language ofrece al lector interesado detallada información acerca de este lenguaje de programación.

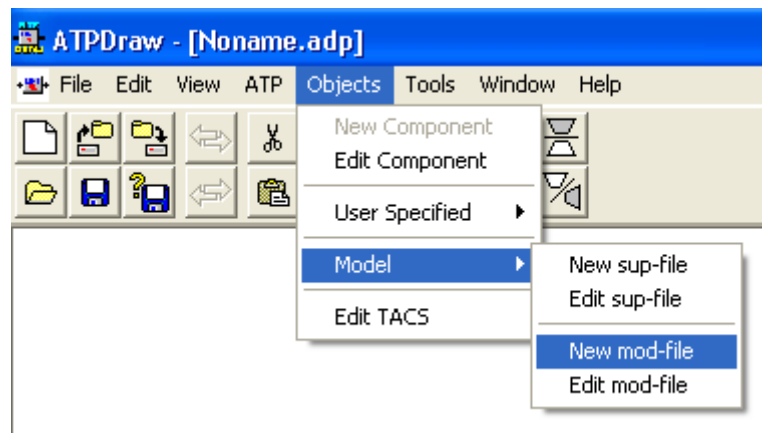


Figura. Opción para crear un modelo mediante MODELS.

Ejemplo de fichero **.mod**:

MODEL FLASH_1

```

comment *****
*
*   Function: set or cancel the gap firing control signal
*   Inputs  : voltage and current across ZnO resistor
*   Output  : the firing signal to the electrical ZnO component
*
***** endcomment

INPUT  V1      -- Voltage on positive side of ZNO      [V]
        V2      -- Voltage of negative side of ZNO    [V]
        iczn    -- ZNO current                        [Amps]

DATA   Pset     -- power setting                       [Megajoules/msec]
        Eset     -- energy setting                     [Megajoules]
        fdel     -- firing delay                       [msec]
        fdur     -- firing duration                     [msec]

VAR    power    -- power into ZnO resistor              [Watts]
        trip     -- gap firing control signal           [0 or 1]
        energy   -- energy into ZnO resistor            [Joules]
        tfire    -- time at which the gap was last fired [sec]
        vcap     -- voltage difference across series caps [Volts]

OUTPUT trip

HISTORY INTEGRAL(power) {DFLT:0}

INIT trip:=0
    tfire:=0
ENDINIT

```

EXEC

```
-----
vcap:=V1-V2
power:=vcap*iczn
energy:=INTEGRAL(power)
-----
IF trip>0          -- is already firing
AND t-tfire>fdur*1.e-3 -- has exceeded firing duration
THEN
    trip:=0        -- cancel the firing signal
    tfire:=0       -- null the firing time
ENDIF
-----
IF trip=0          -- is not signaling to fire
AND tfire=0        -- firing condition not yet detected
AND ( power >= Pset * 1.e9 -- power setting exceeded
      OR energy >= Eset * 1.e6 ) -- energy setting exceeded
THEN
    tfire:=t       -- set the firing detection time
ENDIF
-----
IF trip=0          -- is not signaling to fire
AND tfire>0        -- firing condition has been detected
AND t-tfire>fdel*1.e-3 -- firing delay exceeded
THEN
    trip:=1        -- set the firing signal
ENDIF
-----
ENDEXEC
```

ENDMODEL

Igualmente ha de definirse un fichero **.sup**, asociado a dicho modelo, tal y como se mostró anteriormente. El nombre de los nodos ha de coincidir con las variables especificadas en las secciones INPUT y OUTPUT del fichero **.mod**. Ambos ficheros habrán de almacenarse en la carpeta /MOD.

A.8.5 COMPONENTES EN ATPDRAW.

Para mostrar el menú de selección de componentes basta con pinchar en un espacio vacío de la ventana del circuito con el botón derecho del ratón.

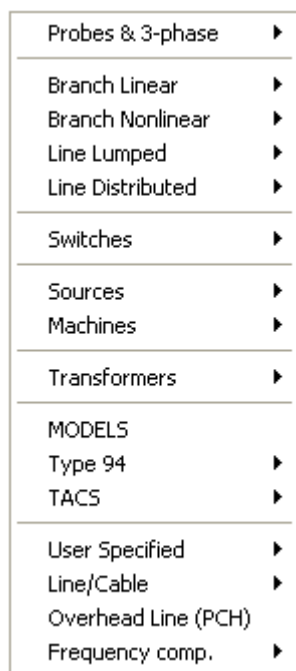


Figura. Menú de selección de componentes.

A.8.5.1 PROBES & 3-PHASE.

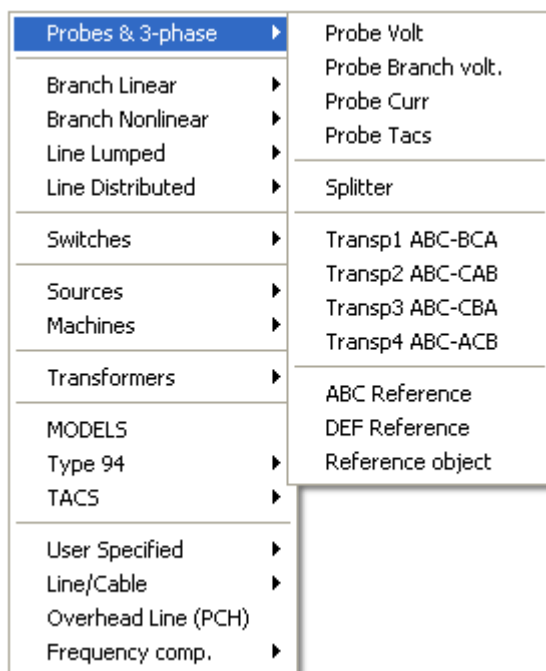


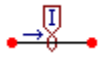
Figura. Menú Probes & 3-phase.

Probe Volt

Mide la tensión a tierra de un nodo.

Probe Branch volt.

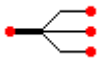
Mide la tensión entre dos nodos.

Probe Curr

Mide la intensidad entre dos nodos.

Probe Tacs

Mide una señal de tipo TACS.

Splitter

Permite la transformación de un nodo monofásico a un nodo trifásico.

A.8.5.2 BRANCH LINEAR.

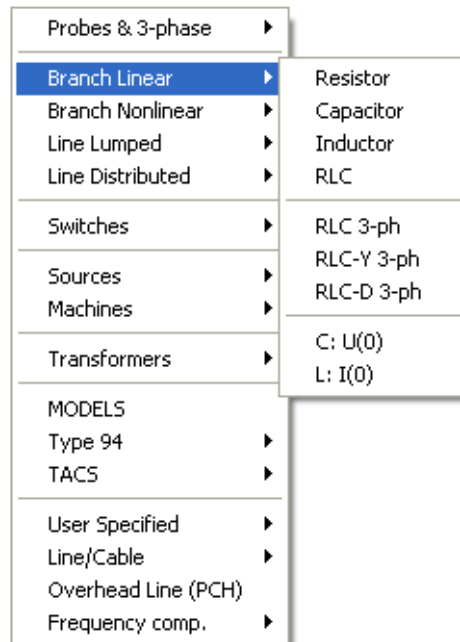


Figura. Menú Branch Linear.

Selection	Object name	Icon	ATP card	Description
<i>Resistor</i>	RESISTOR		BRANCH type 0	Pure resistance in Ω .
<i>Capacitor</i>	CAP_RS		BRANCH type 0	Capacitor with damping resistor. C in μF if Copt=0.
<i>Inductor</i>	IND_RP		BRANCH type 0	Inductor with damping resistor. Inductance in mH if Xopt=0.
<i>RLC</i>	RLC		BRANCH type 0	R, L and C in series.
<i>RLC 3-ph</i>	RLC3		BRANCH type 0	3-phase R, L and C in series. Independent values in phases.
<i>RLC-Y 3-ph</i>	RLCY3		BRANCH type 0	3-phase R, L and C, Y coupling. Independent values in phases.
<i>RLC-D 3-ph</i>	RLCD3		BRANCH type 0	3-phase R, L and C, D coupling. Independent values in phases.
<i>C : U(0)</i>	CAP_U0		BRANCH + initial condition	Capacitor with initial condition.
<i>L : I(0)</i>	IND_I0		BRANCH + initial condition	Inductor with initial condition.

Tabla. Branch linear.

A.8.5.3 BRANCH NONLINEAR.

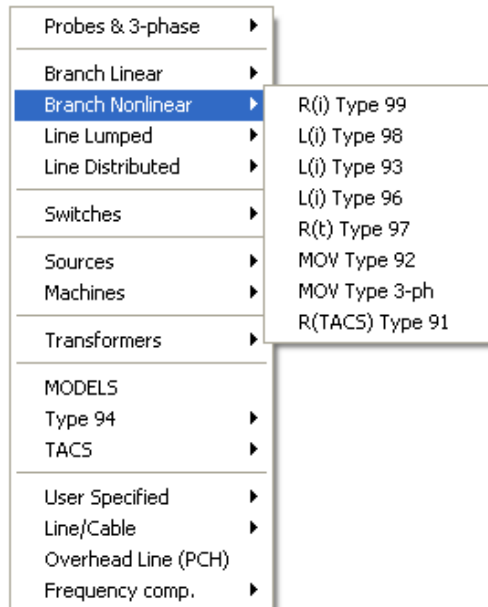


Figura. Menú Branch Nonlinear.

Selection	Object name	Icon	ATP card	Description
<i>R(i) Type 99</i>	NLINRES		BRANCH type 99	Current dependent resistance.
<i>L(i) Type 98</i>	NLININD		BRANCH type 98	Current dependent inductor.
<i>L(i) Type 93</i>	NLIND93		BRANCH type 93	True non-linear current dependent inductor.
<i>L(i) Type 96</i>	NLIND96		BRANCH type 96	Pseudo-nonlinear hysteretic inductor.
<i>R(t) Type 97</i>	NLINR_T		BRANCH type 97	Time dependent resistor.
<i>MOV Type 92</i>	MOV		BRANCH type 92	Current dependent resistance on exponential form.
<i>MOV Type 3-ph</i>	MOV_3		BRANCH type 92	3-phase current dependent resistance.
<i>R(TACS) Type 91</i>	TACSRES		BRANCH type 91	TACS / MODELS controlled time dependent resistor.

Tabla. Branch Nonlinear.

A.8.5.4 LINE LUMPED.

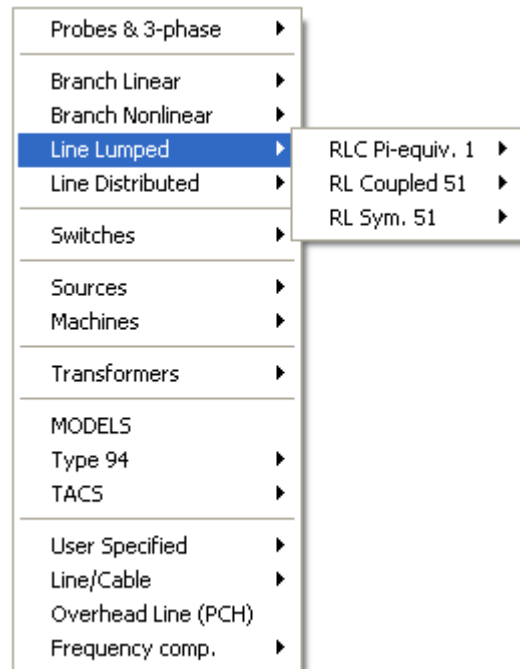


Figura. Menú Line Lumped.

Selection	Object name	Icon	ATP card	Description
<i>RLC Pi-equiv. 1 + 1 phase</i>	LINEPI_1		BRANCH type 1	Single phase RLC π -equivalent.
<i>RLC Pi-equiv. 1 + 2 phase</i>	LINEPI_2		BRANCH type 1-2	2-phase RLC π -equivalent Non-symmetric.
<i>RLC Pi-equiv. 1 + 3 phase</i>	LINEPI_3		BRANCH type 1-3	3-phase RLC π -equivalent Non-symmetric. 3-phase nodes.
<i>RL Coupled 51 + 2 phase</i>	LINERL_2		BRANCH type 51-52	2-phase RL coupled line model. Non-symmetric.
<i>RL Coupled 51.. + 3 phase</i>	LINERL_3		BRANCH type 51-53	3-phase RL coupled line model. Non-symmetric. 3-phase nodes.
<i>RL Coupled 51.. + 6 phase</i>	LINERL_6		BRANCH type 51-56	2x3 phase RL coupled line model. Non-symmetric. Off- diagonal R is set to zero.
<i>RL Sym. 51 + 3 ph</i>	LINESY_3		BRANCH type 51-53	3-phase RL coupled line model with sequence impedance (0, +) input. Symmetric.
<i>RL Sym. 51 + 6 ph</i>	LINESY_6		BRANCH type 51-56	2x3-phase RL coupled line model with sequence impedance (0, +) input. Symmetric.

Tabla. Line Lumped.

A.8.5.5 LINE DISTRIBUTED.

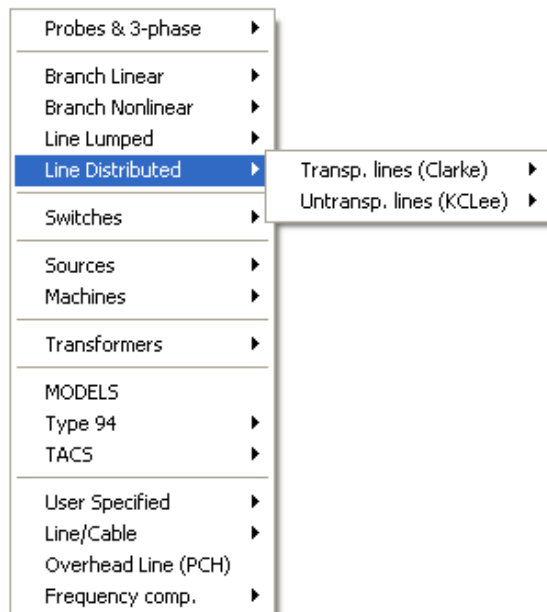


Figura. Menú Line Distributed.

Selection	Object name	Icon	ATP card	Description
Transposed lines + 1 phase	LINEZT_1		BRANCH type -1	Single phase, distributed parameter line, Clarke model.
Transposed lines + 2 phase	LINEZT_2		BRANCH type -1.. -2	2-phase, distributed parameter, transposed line, Clarke model.
Transposed lines + 3 phase	LINEZT_3		BRANCH type -1.. -3	3-phase, distributed parameter, transposed line, Clarke model.
Transposed lines + 6 phase	LINEZT6N		BRANCH type -1.. -6	6-phase, distributed parameter, transposed line, Clarke model.
Transposed lines + 6 phase mutual	LINEZT_6		BRANCH type -1.. -6	2x3 phase, distributed Clarke line. with mutual coupling between the circuits.
Transposed lines + 9 phase	LINEZT_9		BRANCH type -1.. -9	9-phase, distributed parameter, transposed line, Clarke model.
Untransposed lines (KCLee)+ 2 phase	LINEZU_2		BRANCH	2-phase, distributed parameters, untransposed (KCLee) line model with complex transformation matrix.
Untransposed lines (KCLee)+ 3 phase	LINEZU_3		BRANCH	3-phase, distributed parameters, untransposed (KCLee) line model with complex transformation matrix

abla. Line Distributed.

A.8.5.6 SWITCHES.

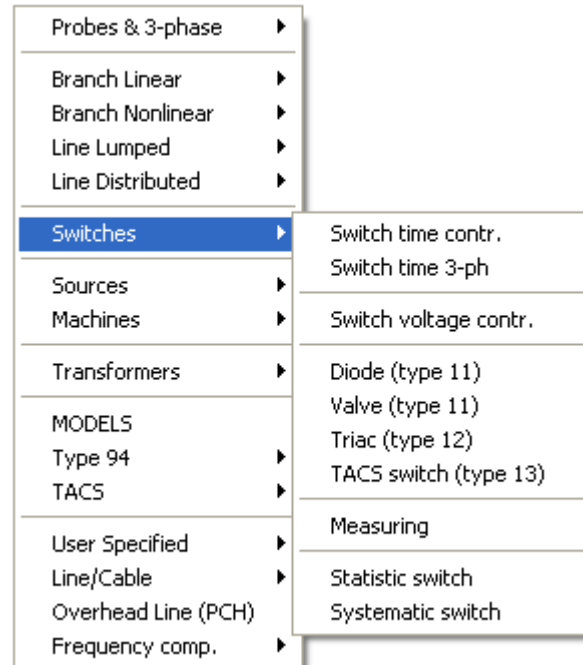


Figura. Menú Switches.

Selection	Object name	Icon	ATP card	Description
<i>Switch time controlled</i>	SWITCHTC		SWITCH type 0	Single phase time controlled switch.
<i>Switch time 3-ph</i>	SWIT_3XT		SWITCH type 0	Three-phase time controlled switch, Independent operation of phases.
<i>Switch voltage contr.</i>	SWITCHVC		SWITCH type 0	Voltage controlled switch.
<i>Diode (type 11)</i>	DIODE		SWITCH type 11	Diode. Switch type 11. Uncontrolled.
<i>Valve (type 11)</i>	VALVE		SWITCH type 11	Valve/Thyristor. Switch type 11. TACS/MODELS- controlled.
<i>Triac (type 12)</i>	TRIAC		SWITCH type 12	Double TACS/MODELS controlled switch.
<i>TACS switch (type 13)</i>	TACSSWIT		SWITCH type 13	Simple TACS/MODELS controlled switch.
<i>Measuring</i>	SWMEAS		SWITCH type 0	Measuring switch. Current measurements.
<i>Statistic switch</i>	SW_STAT		SWITCH	Statistic switch. See ATP Settings / Switch/UM.
<i>Systematic switch</i>	SW_SYST		SWITCH	Systematic switch. See ATP Settings / Switch/UM.

Tabla. Switches.

A.8.5.7 SOURCES.

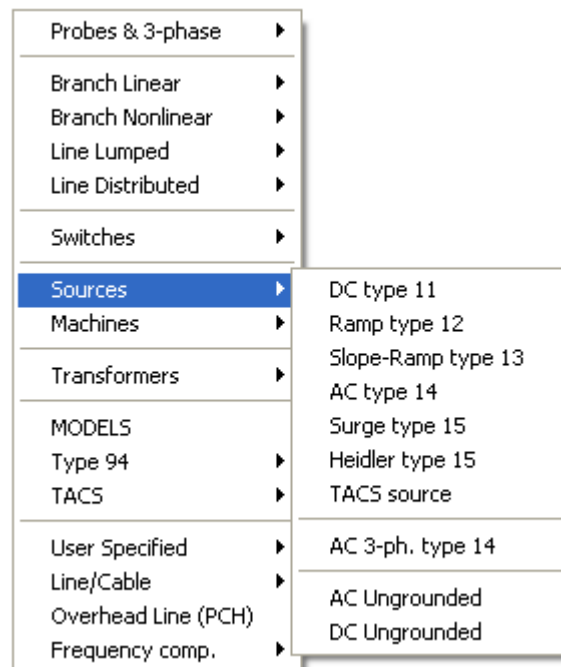


Figura. Menú Sources.

Selection	Object name	Icon	ATP card	Description
<i>DC type 11</i>	DC1PH		SOURCE type 11	DC step source. Current or voltage.
<i>Ramp type 12</i>	RAMP		SOURCE	Ramp source. Current or voltage.
<i>Slope-Ramp type 13</i>	SLOPE_RA		SOURCE type 13	Two-slope ramp source. Current or voltage.
<i>AC type 14</i>	AC1PH		SOURCE	AC source. Current or voltage.
<i>Surge type 15</i>	SURGE		SOURCE	Double exponential source Type-15.
<i>Heidler type 15</i>	HEIDLER		SOURCE	Heidler type source. Current or Volt.
<i>TACS source</i>	TACSSOUR		SOURCE type 60	TACS/MODELS controlled source. Current or voltage.
<i>AC 3-ph. type 14</i>	AC3PH		SOURCE type 14	AC source. Current or voltage. 3-phase node.
<i>AC Ungrounded</i>	AC1PHUG		SOURCE type 14+18	Ungrounded AC source. Voltage only.
<i>DC Ungrounded</i>	DC1PHUG		SOURCE type 11+18	Ungrounded DC source. Voltage only.

Tabla. Sources.

A.8.5.8 MACHINES.

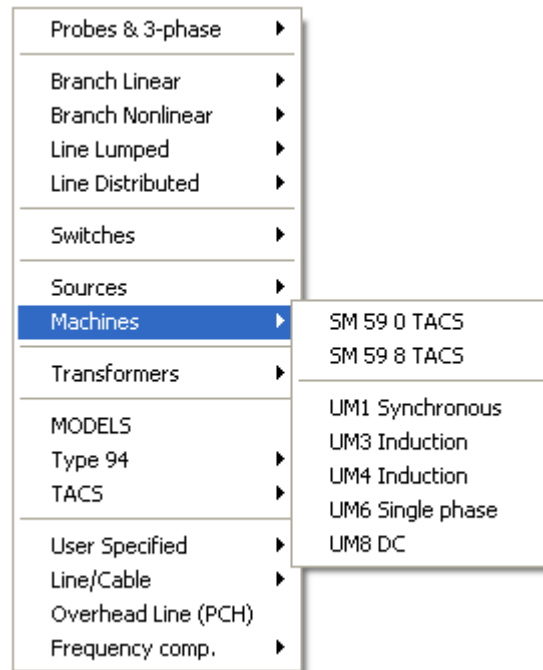


Figura. Menú machines.

Selection	Object name	Icon	ATP card	Description
<i>SM 59</i> + No control	SM59_NC		MACHINE type 59	Synchronous machine. No TACS control. 3-phase armature.
<i>SM 59</i> + 8 control	SM59_FC		MACHINE type 59	Synchronous machine. Max. 8 TACS control. 3-phase armature.
<i>UM1</i> Synchronous	UM_1		UM-MACHINE type 1	Synchronous. Set initialization under <i>ATP Settings/Switch/UM.</i>
<i>UM3</i> Induction	UM_3		UM-MACHINE type 3	Induction. Set initialization under <i>ATP Settings/Switch/UM.</i>
<i>UM4</i> Induction	UM_4		UM-MACHINE type 4	Induction. Set initialization under <i>ATP Settings/Switch/UM.</i>
<i>UM6 Single</i> <i>phase</i>	UM_6		UM-MACHINE type 6	Single phase. Set initialization under <i>ATP Settings/Switch/UM.</i>
<i>UM8 DC</i>	UM_8		UM-MACHINE type 8	DC machine. Set initialization under <i>ATP Settings/Switch/UM.</i>

Tabla. Machines.

A.8.5.9 TRANSFORMERS.

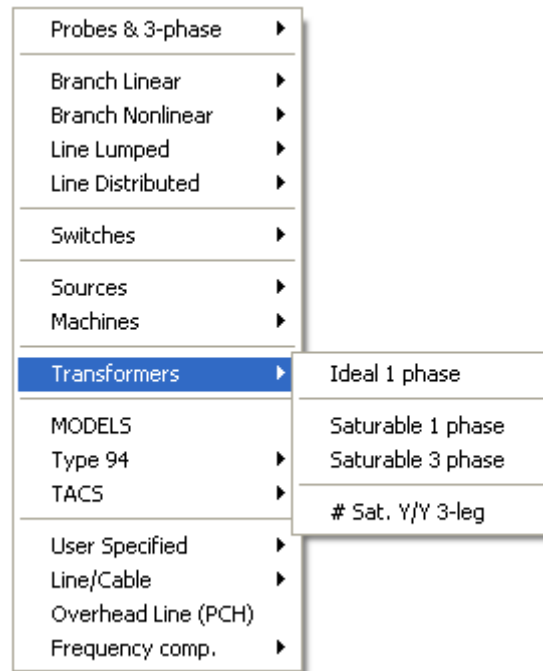
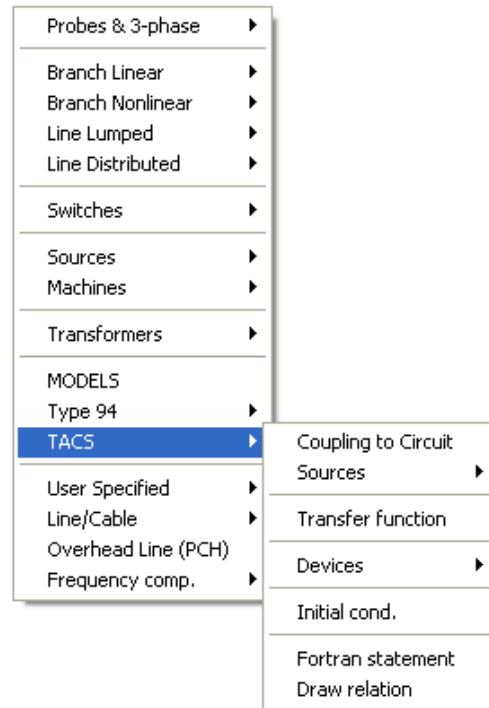


Figure. Menú Transformers.

Selection	Object name	Icon	ATP card	Description
<i>Ideal 1 phase</i>	TRAFO_I		SOURCE type 18	Single-phase ideal transformer.
<i>Ideal 3 phase</i>	TRAFO_I3		SOURCE type 18	3-phase ideal transformer.
<i>Saturable 1 phase</i>	TRAFO_S		BRANCH TRANSFORMER	Single-phase saturable transformer.
<i>Saturable 3 phase</i>	GENTRAFO		BRANCH TRANSFORMER	General saturable transformer. 3-phase. 2 or 3 windings.
<i># Sat. Y/Y 3-leg</i>	TRAYH_3		BRANCH TRANSFORMER THREE PHASE	3-phase saturable transformer. High homopolar reluct. (3-leg). 3-ph node. Preprocessing of manufacturer data.
<i>BCTAN</i>	BCTAN		BRANCH Type 1...9	Direct support of BCTAN transformer matrix modeling.

Tabla. Transformers.

A.8.5.10 TACS.



Coupling to circuit

Selection	Object name	Icon	ATP card	Description
<i>Coupling to Circuit</i>	EMTP_OUT		TACS type 90-93	Value from the electrical circuit into TACS. 90 - Node voltage 91 - Switch current 92 - internal variable special EMTP comp. 93 - Switch status.

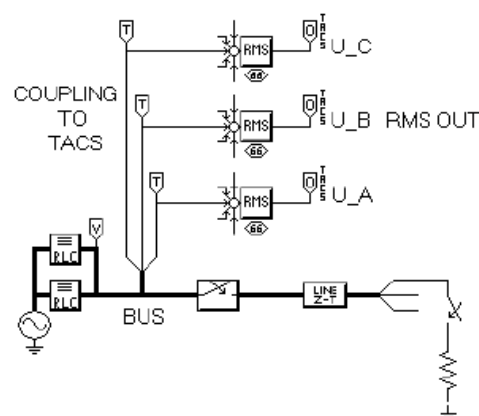


Figura. Acoplamiento de un nodo eléctrico trifásico a TACS.

TACS sources

Selection	Object name	Icon	ATP card	Description
<i>DC - 11</i>	DC_01		TACS type 11	TACS step signal source.
<i>AC - 14</i>	AC_02		TACS type 14	TACS AC cosine signal source.
<i>Pulse - 23</i>	PULSE_03		TACS type 23	TACS pulse train signal.
<i>Ramp - 24</i>	RAMP_04		TACS type 24	TACS saw-tooth train signal.

Tabla. TACS sources.

TACS transfer function

Selection	Object name	Icon	ATP card	Description
<i>General</i>	TRANSF		TACS	General transfer function in s domain. Order 0-7. Named dynamic limits.

Tabla. TACS transfer function.

Fortran statement

Selection	Object name	Icon	ATP card	Description
<i>General</i>	FORTTRAN1		TACS type 88,98 or 99	User specified FORTRAN expression.

Tabla. Fortran statement.

TACS devices

Selection	Object name	Icon	ATP card	Description
<i>Freq sensor - 50</i>	DEVICE50		TACS type 88,98 or 99	Frequency sensor.
<i>Relay switch - 51</i>	DEVICE51		TACS type 88,98 or 99	Relay-operated switch.
<i>Level switch - 52</i>	DEVICE52		TACS type 88,98 or 99	Level-triggered switch.
<i>Trans delay - 53</i>	DEVICE53		TACS type 88,98 or 99	Transport delay.
<i>Pulse delay - 54</i>	DEVICE54		TACS type 88,98 or 99	Pulse delay.
<i>User def nonlin - 56</i>	DEVICE56		TACS type 88,98 or 99	Point-by-point non-linearity.
<i>Multi switch - 57</i>	DEVICE57		TACS	Multiple open/close switch.
<i>Cont integ - 58</i>	DEVICE58		TACS type 88,98 or 99	Controlled integrator.
<i>Simple deriv - 59</i>	DEVICE59		TACS type 88,98 or 99	Simple derivative.
<i>Input IF - 60</i>	DEVICE60		TACS type 88,98 or 99	Input-IF component.
<i>Signal select - 61</i>	DEVICE61		TACS type 88,98 or 99	Signal selector.
<i>Sample_track - 62</i>	DEVICE62		TACS type 88,98 or 99	Sample and track.
<i>Inst min/max - 63</i>	DEVICE63		TACS type 88,98 or 99	Instantaneous minimum/maximum.
<i>Min/max track - 64</i>	DEVICE64		TACS type 88,98 or 99	Minimum/maximum tracking.
<i>Acc count - 65</i>	DEVICE65		TACS type 88,98 or 99	Accumulator and counter.
<i>Rms meter - 66</i>	DEVICE66		TACS type 88,98 or 99	RMS value of the sum of input signals.

Tabla. TACS devices.

A.8.5.11 Frequency comp.

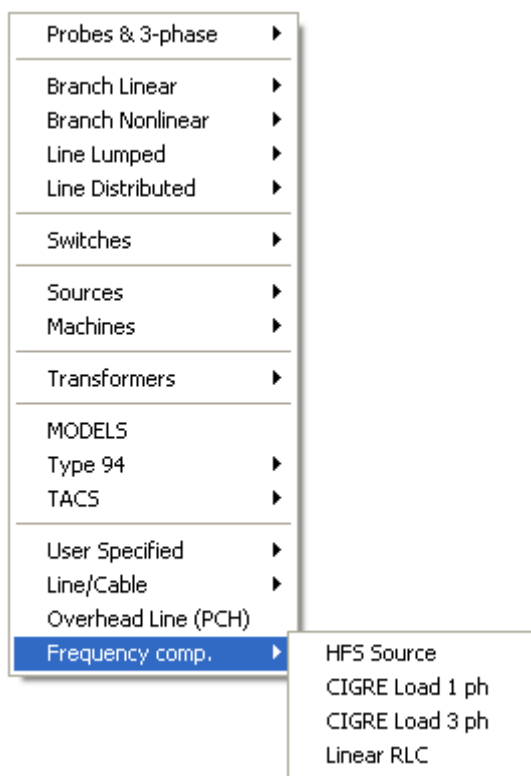


Figura. Menú Frequency comp.

Selection	Object name	Icon	ATP card	Description
<i>HFS Source</i>	HFS_SOUR		SOURCE type 14	Harmonic frequency source
<i>Cigre load 1 ph</i>	CIGRE_1		BRANCH type 0	Single-phase CIGRE load
<i>Cigre load 3 ph</i>	CIGRE_3		BRANCH type 0	3-phase CIGRE load
<i>Linear RLC</i>	RLC_F		BRANCH type 0	Linear RLC for HFS studies

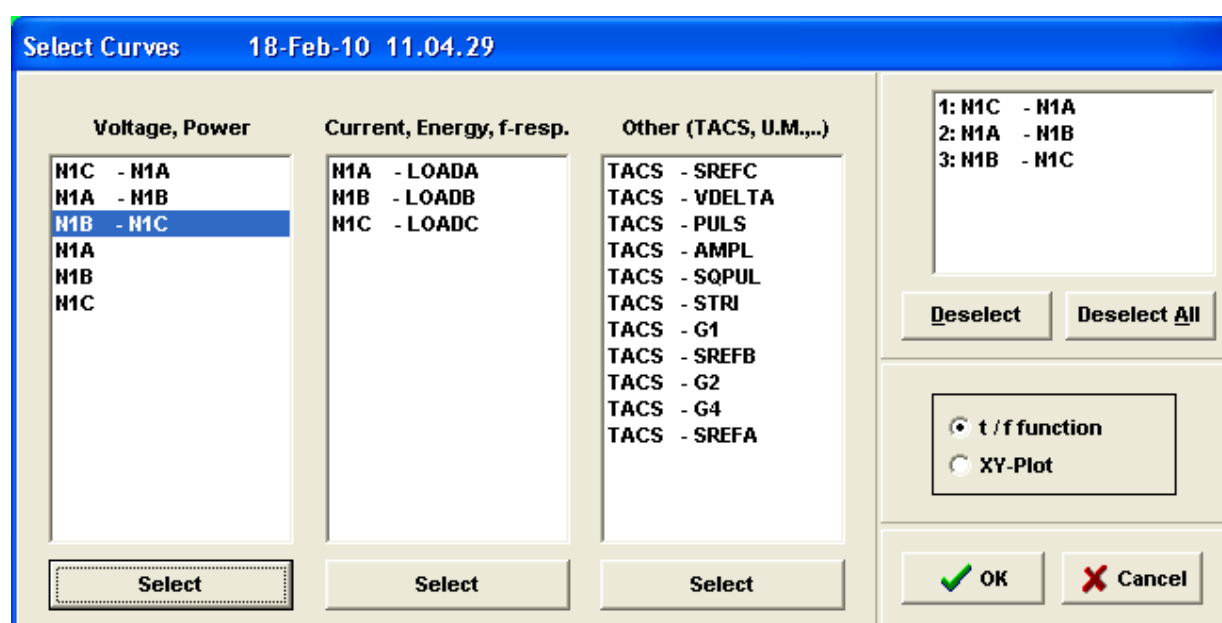
Tabla. Frequency comp.

A.9 Visualizador gráfico: PCPlot.

El programa PCPlot permite la visualización interactiva de los resultados guardados en los ficheros de extensión .pl4 que genera ATP.

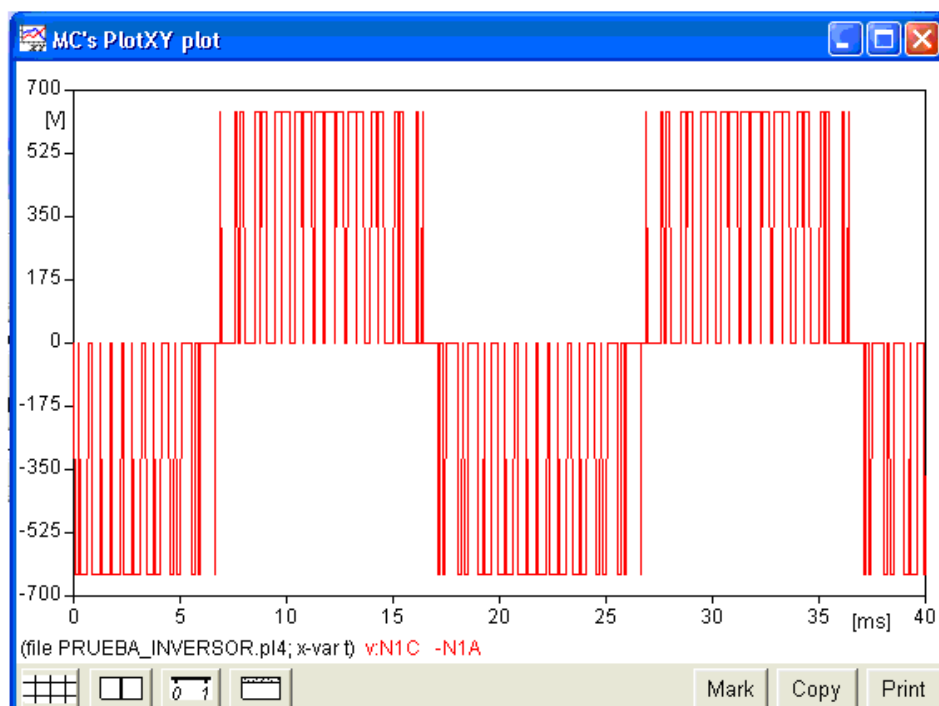
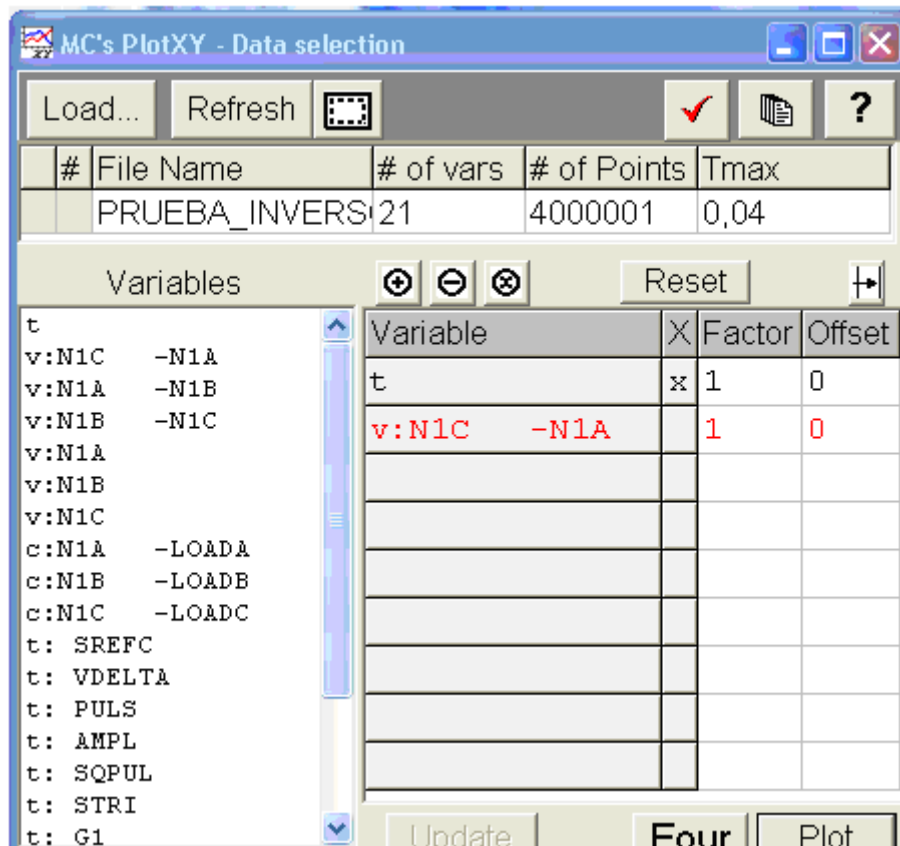
Al abrir un nuevo fichero aparece la ventana de diálogo de la figura, donde se eligen las variables a representar y el tipo de representación deseado.

Las variables escogidas se pueden dibujar en función del tiempo o en función de una de ellas, según se seleccione la opción correspondiente. En el segundo caso, la primera de las variables seleccionada constituye el eje x y aparece señalizada con una 'X' a su izquierda.



A.10 Visualizador gráfico: PlotXY.

Permite representar hasta ocho curvas en la misma gráfica, realizar la transformada de Fourier de una señal, representar en la misma hoja curvas de tres ficheros diferentes, representar las curvas en función del tiempo o hacer una representación X-Y, hacer un escalado automático de ejes, acceder al valor instantáneo de forma numérica, exportar datos, etc.



A.11 Oscilaciones numéricas.

A.11.1 Causas de las oscilaciones numéricas.

El programa EMTP de transitorios electromagnéticos usa el método de integración numérica denominado ‘regla trapezoidal’. La regla trapezoidal es una técnica de integración numérica de segundo orden que es simple de implementar, estable y rápido. Sin embargo, es también susceptible de oscilaciones numéricas cuando se diferencian tensiones o corrientes respecto al tiempo de paso. Ello es común cuando se considera el paso de corriente a través de una inductancia o la tensión entre los terminales de un condensador.

$$\text{Tensión en una inductancia: } v = L \cdot \frac{di}{dt}$$

$$\text{Corriente en un condensador: } i = C \cdot \frac{dv}{dt}$$

Las oscilaciones numéricas se hacen presentes cuando se da una variación de corriente respecto del tiempo demasiado elevada a través de una inductancia o, igualmente, cuando aparece una diferencia de tensión significativamente alta entre los terminales de un condensador. Una posible solución consistiría en la realización de un circuito más exacto, incluyendo condensadores parásitos y resistencias internas.

A.11.2 Representación matemática.

El problema puede ser representado considerando la ecuación

$$x = \frac{dy}{dt}$$

usando la regla trapezoidal. Podemos integrar ambos lados de la ecuación,

$$y(t) = \int_{t-\Delta t}^t x dt + y(t - \Delta t)$$

A continuación reescribimos la ecuación usando la regla trapezoidal. Se ha de hacer notar que estamos buscando una aproximación trapezoidal de la frontera formada por x , t y $t - \Delta t$.

$$y(t) = y(t - \Delta t) + \frac{\Delta t}{2} (x(t) + x(t - \Delta t))$$

Se resuelve por consiguiente esta ecuación para $x(t)$, que puede representar la intensidad a través de un condensador o la diferencia de potencial entre los terminales de una inductancia.

$$x(t) = -x(t - \Delta t) + \frac{2}{\Delta t}(y(t) - y(t - \Delta t))$$

El resultado esperado habría de ser el mostrado en la siguiente figura:

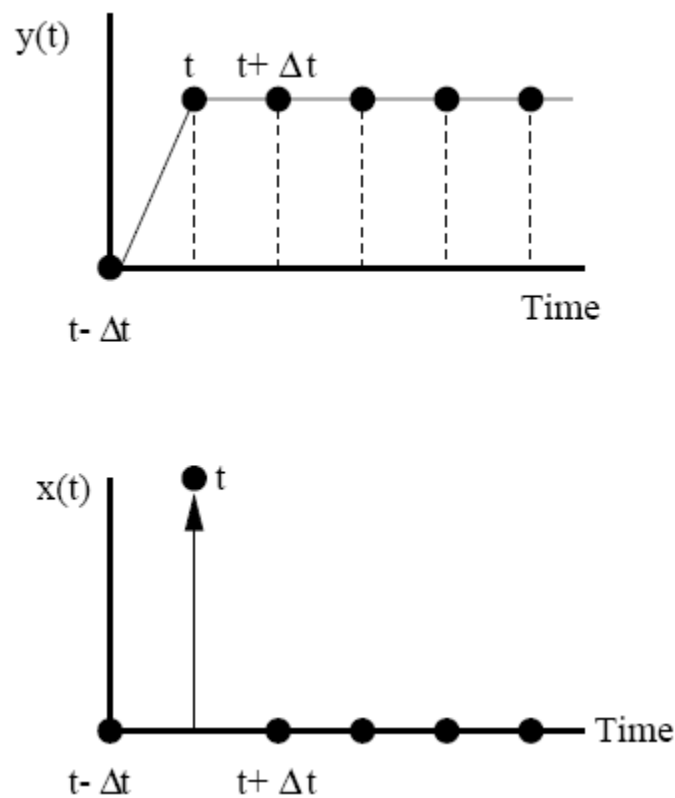


Figura. Representación esperada de $x(t)$ e $y(t)$ para el caso de ejemplo.

Así, para el primer paso de tiempo se obtiene:

$$x(t) = \frac{2}{\Delta t}$$

El siguiente paso, $x(t-\Delta t)$, sería:

$$x(t + \Delta t) = -\frac{2}{\Delta t} + \frac{2}{\Delta t}(1.0 - 1.0) = -\frac{2}{\Delta t}$$

Un nuevo paso, sin embargo, resultaría:

$$x(t + 2\Delta t) = \frac{2}{\Delta t}$$

Se aprecia en este caso la existencia de una oscilación, como aparece en la siguiente figura. El valor medio de $x(t)$ corresponde a la forma de onda deseada.

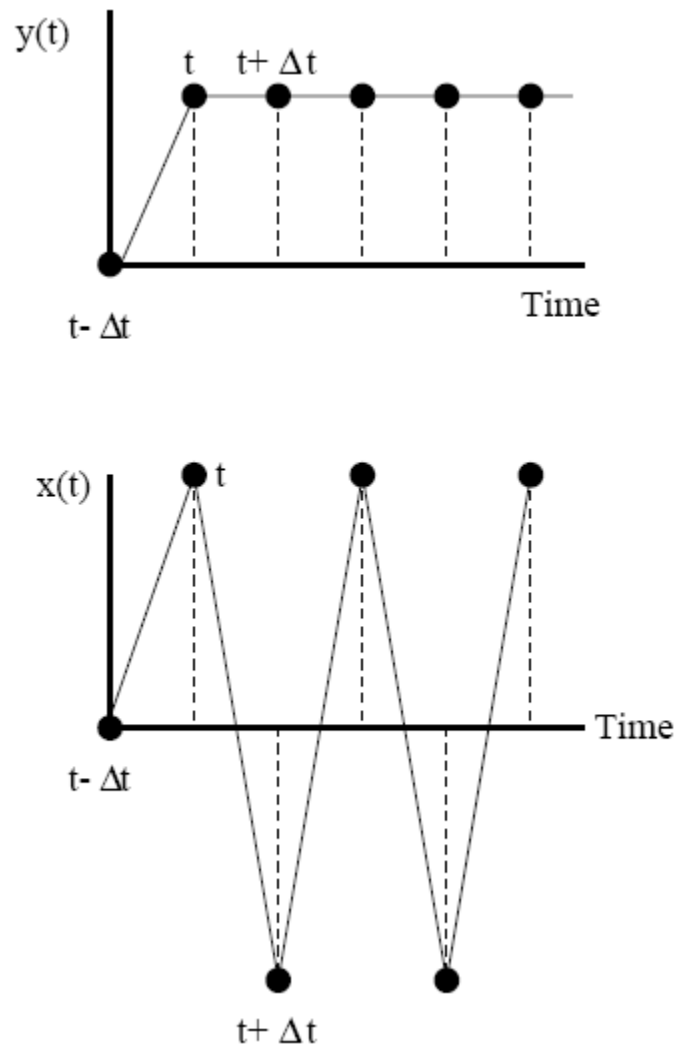


Figura. Actual $x(t)$ con oscilaciones numéricas.

Una transición de $y(t)$ de dos pasos de tiempo en lugar de uno resolvería esta situación, como se muestra a través del siguiente conjunto de ecuaciones. Se asume en este caso $y(t-\Delta t)=0$, $y(t)=0.5$ e $y(t+\Delta t)=1.0$.

$$x(t) = 0.5 \cdot \frac{2}{\Delta t} = \frac{1}{\Delta t}$$

$$x(t + \Delta t) = -0.5 \cdot \frac{2}{\Delta t} + \frac{2}{\Delta t} (1.0 - 0.5) = 0.0$$

$$x(t + 2\Delta t) = 0.0 + \frac{2}{\Delta t} (1.0 - 1.0) = 0.0$$

$$x(t + 3\Delta t) = 0.0$$

...

En la siguiente figura se ilustra los valores de $x(t)$ e $y(t)$ en consideración de estas ecuaciones:

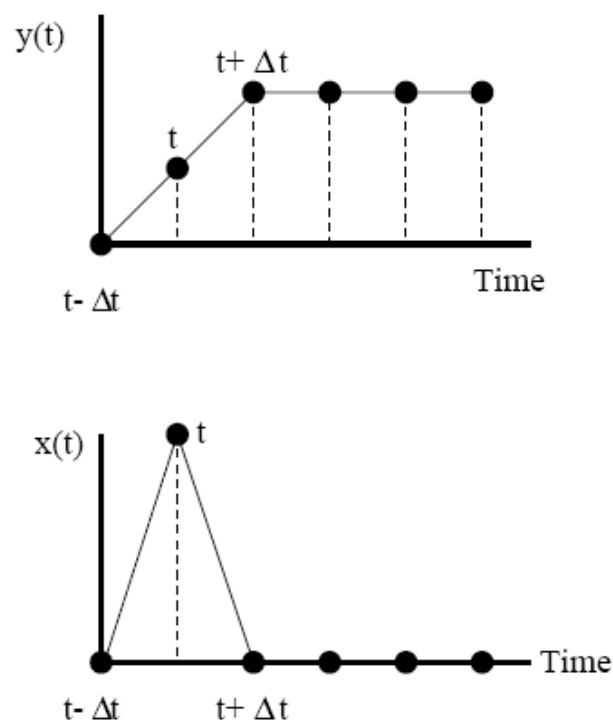


Figura. Representación de $y(t)$ y $x(t)$ con tiempo de subida más lento.

De todo ello, se deduce que reducir el paso de tiempo permite en algunos casos paliar o eliminar la aparición de oscilaciones numéricas. Se debe señalar sin embargo que la aparición de oscilaciones debidas a operaciones de conmutación no se verán reducidas de esta manera, ya que el interruptor ideal siempre cambiará en un paso de tiempo.

A.11.3 Ejemplos.

A.11.3.1 Intensidad a través de una inductancia.

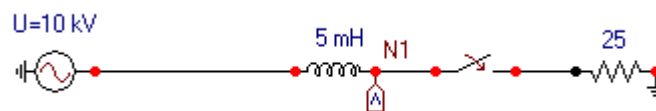


Figura. Circuito de ejemplo.

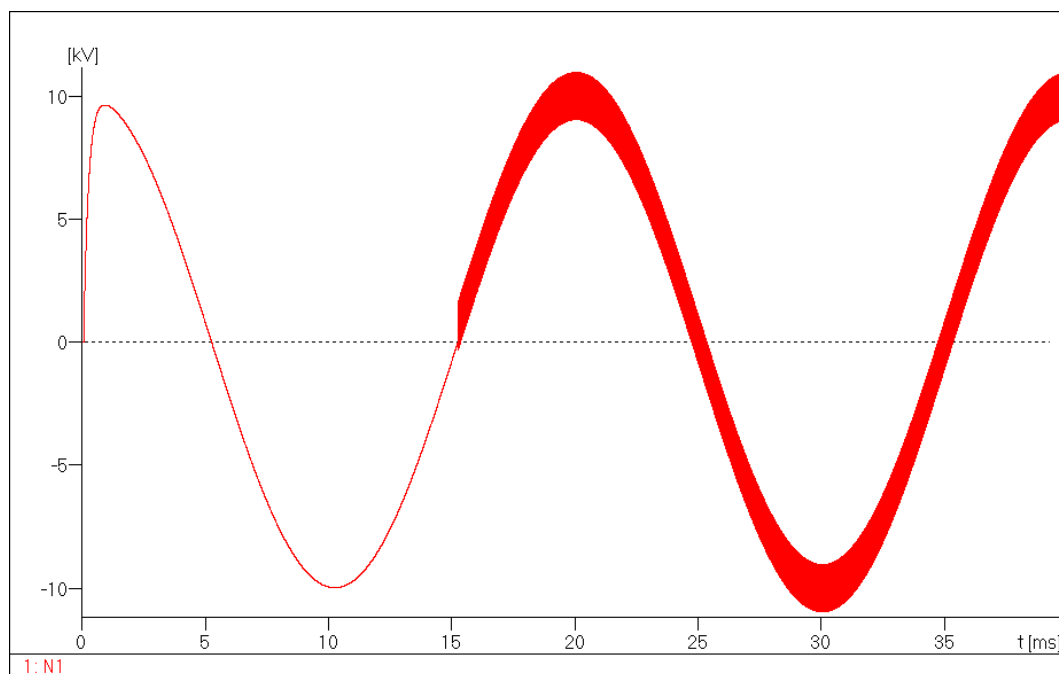


Figura. Tensión en el nodo 1.

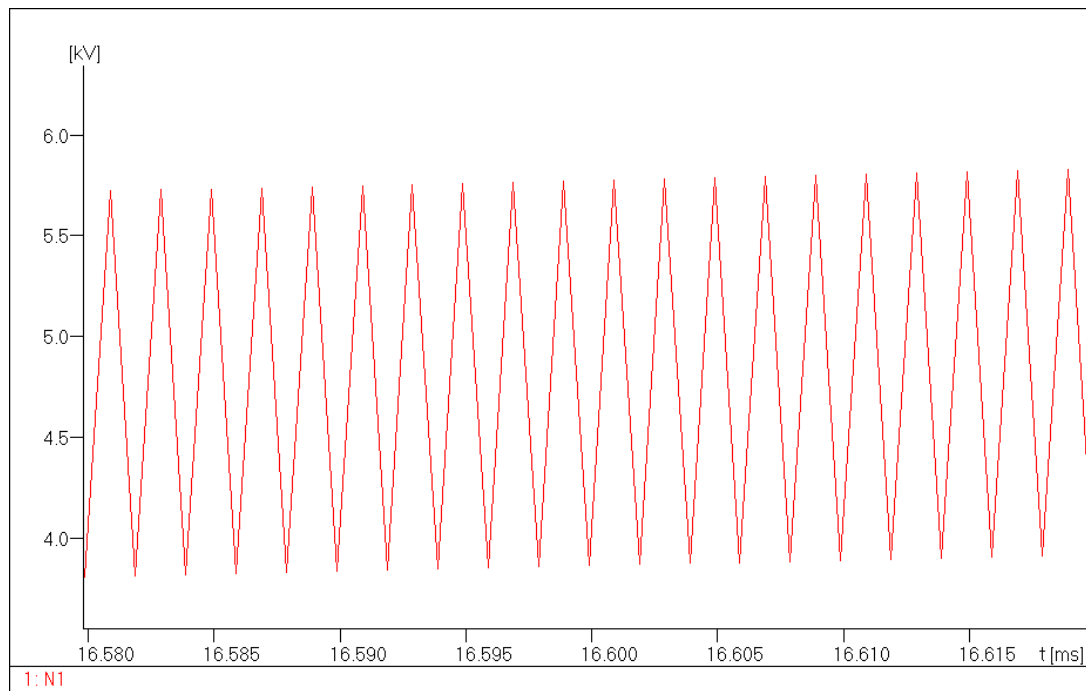


Figura. Zoom de la tensión en el nodo 1.

A.11.3.2 Tensión a través de un condensador.

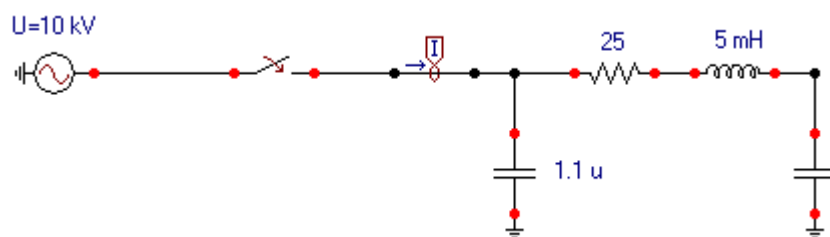


Figura. Circuito de ejemplo.

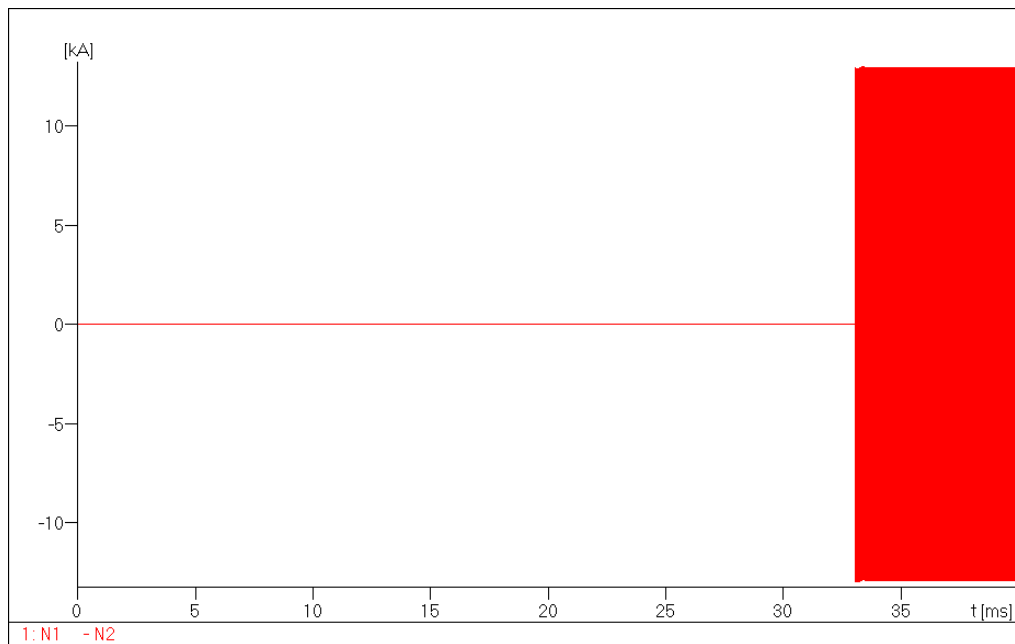


Figura. Intensidad suministrada al circuito pi.

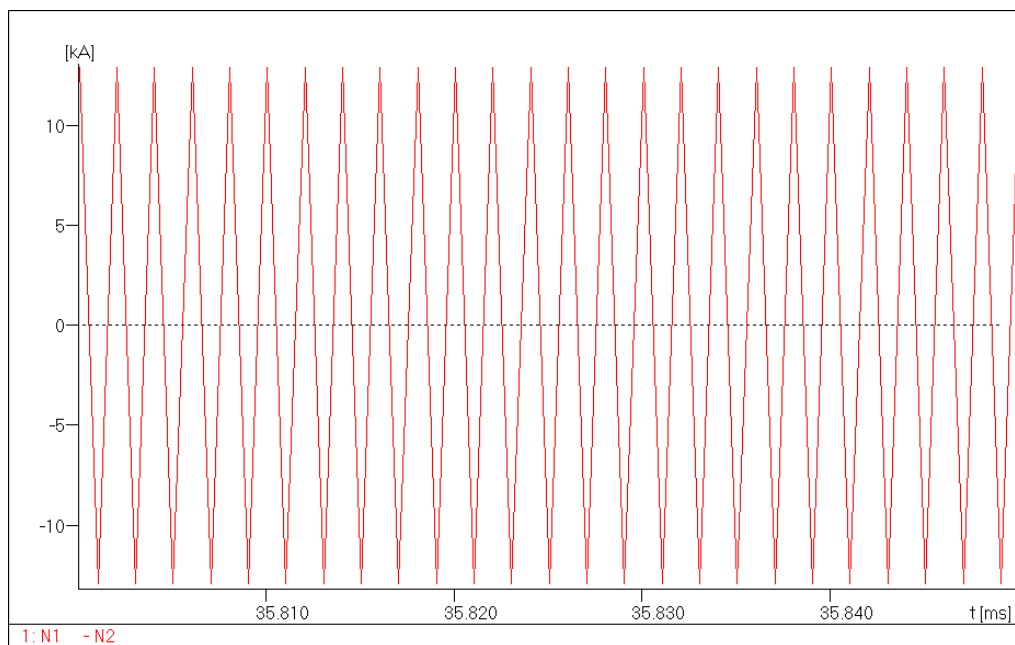


Figura. Zoom de la intensidad suministrada al circuito pi.

A.11.4 Opciones para eliminar las oscilaciones numéricas.

A.11.4.1 Añadiendo elementos al circuito.

El modelo del sistema eléctrico puede carecer de ciertos detalles que den lugar a oscilaciones numéricas. Por ejemplo, podría tenerse en cuenta la impedancia de arco de un interruptor en su apertura. Sin embargo, modelar una impedancia de arco es claramente complejo. En este sentido, puede optarse por incluir un condensador parásito como se ilustra en la siguiente figura. En lugar de oscilaciones numéricas tendrán lugar oscilaciones LC que podrán ocurrir en la práctica.

Igualmente, la diferencia de potencial acaecida entre los terminales de un condensador obvia la resistencia serie efectiva del condensador. Su consideración introduce un término de amortiguamiento.

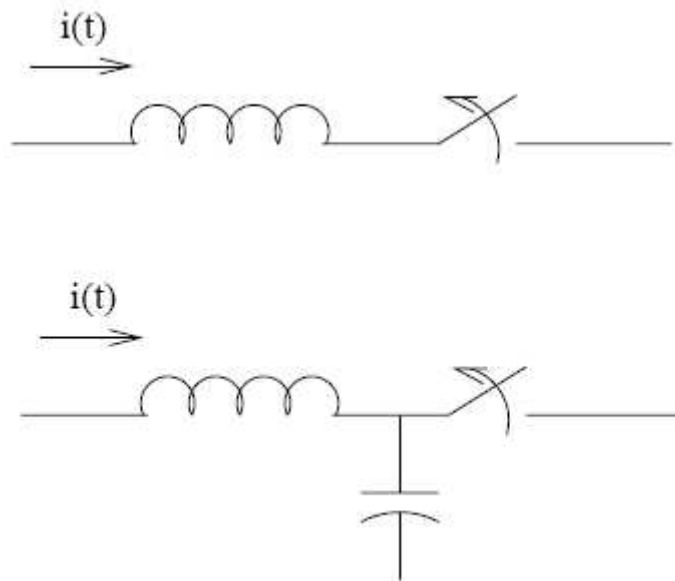


Figura. Interruptor en serie sin y con un condensador parásito.

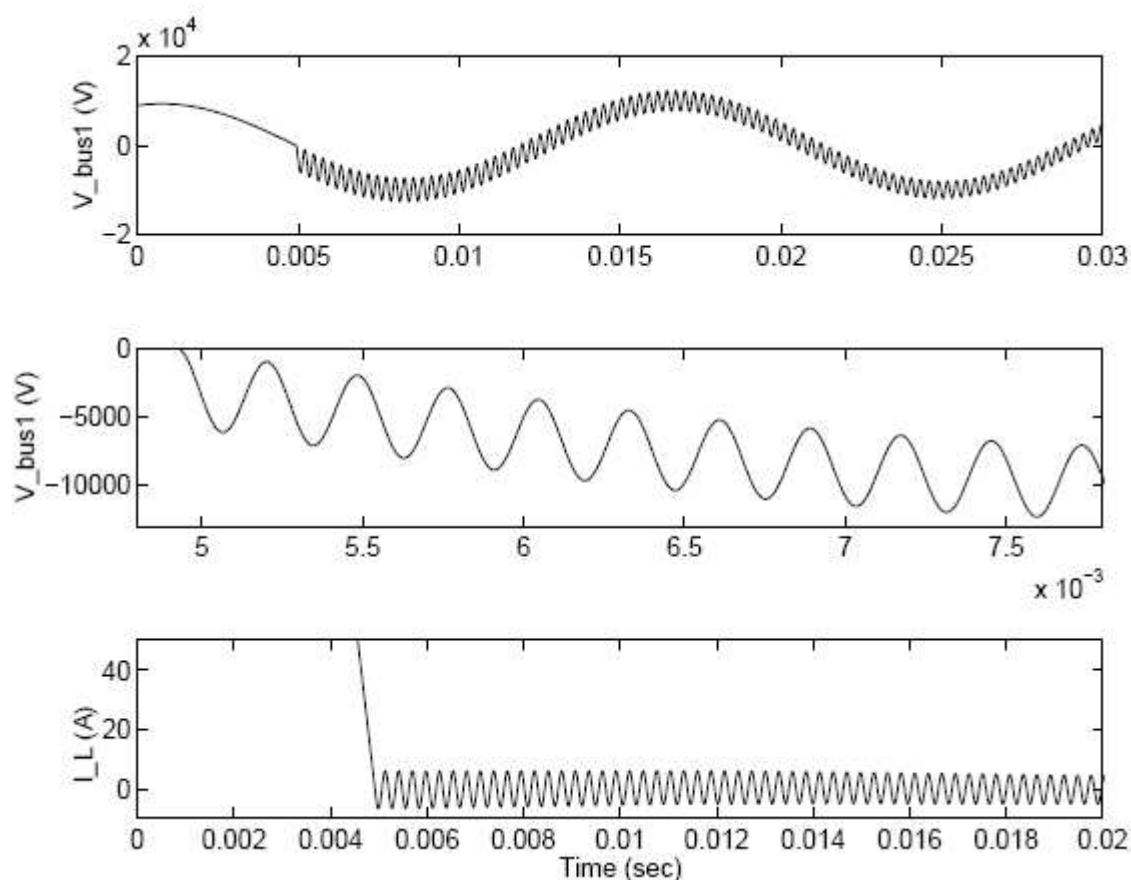


Figura. Simulación con un condensador parásito añadido.

A.11.4.2 Rama ‘snubber’.

Sin embargo, no siempre es práctico introducir estos elementos parásitos adicionales. Este es el caso de ciertos dispositivos electrónicos de potencia. En ellos, la rama paralela o ‘snubber’, proporciona un camino alternativo a la gran corriente inductiva generada por la carga, teniéndose así un circuito RLC de amortiguamiento. La corriente decrece a cero cuando el condensador está cargado. De esta manera, además, se eliminan o reducen los problemas relacionados con oscilaciones numéricas.

Interruptores electrónicos de potencias con grandes áreas de seguridad (*Safe Operating Area*, SOA) tales como IGBTs y MOSFETs no necesitan ‘snubbers’. En estos casos el usuario del programa puede añadir ‘snubbers’ numéricos. El valor del condensador puede estar comprendido entre 1 y 2 nF, si no se considera la resistencia. Por el contrario, si se considera una rama RC ‘snubber’, como se muestra en la figura siguiente, la constante de tiempo $\tau = R \cdot C$ habrá de ser 2 ó 3 veces el tiempo de paso elegido para la simulación. Nótese, sin embargo, que los valores de R y C podrán variar según el sistema eléctrico a simular.

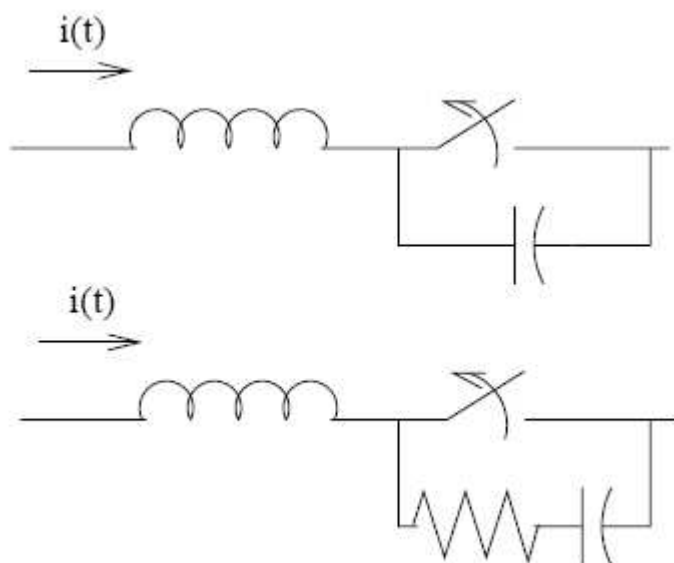


Figura. Opciones 'snubber'.

Llegados a este punto se considera la simulación del paso de corriente a través de una inductancia presentado anteriormente teniendo en cuenta una rama 'snubber'. En la siguiente figura se muestra los resultados de tensión e intensidad obtenidos, donde se aprecia la ausencia de oscilaciones.

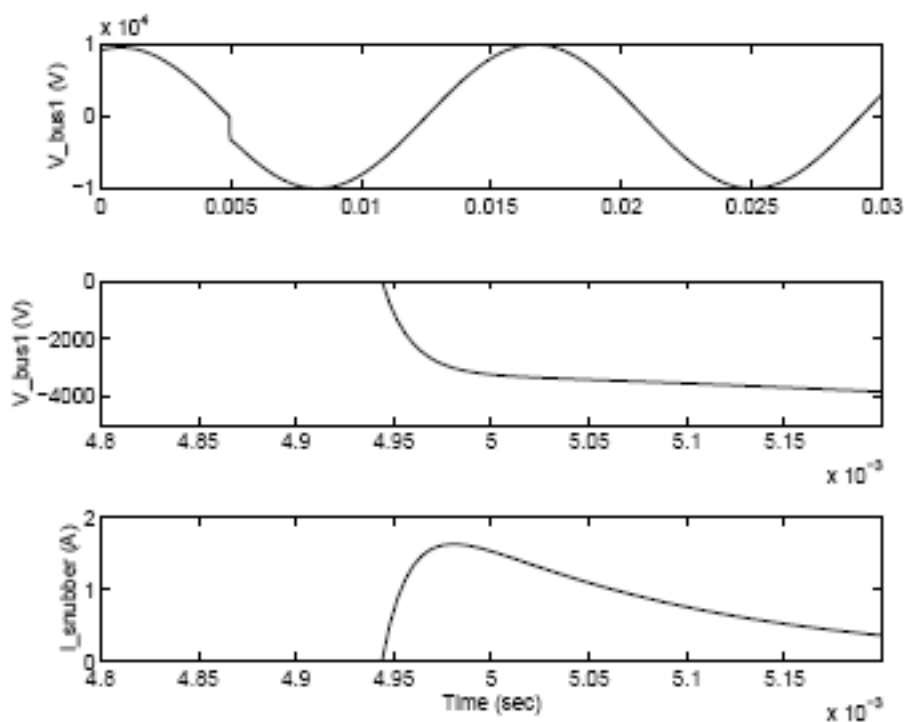


Figura. Ejemplo de la corriente a través de la inductancia con 'snubber' añadida.

A.11.4.3 Disminuir el tiempo de paso.

Un tiempo de paso pequeño puede eliminar o paliar en algunos casos las oscilaciones numéricas presentes en la simulación. Ello depende de la topología del circuito y de la presencia de resistencias en el mismo. Se ha de señalar que el tiempo de paso requerido puede estar lejos de una simulación práctica. Además, debido a que la amplitud de la oscilación varía con $1/\Delta t$, un tiempo de paso pequeño puede aumentar la misma.

A.11.4.4 Amortiguamiento.

Alternativamente, puede considerarse la introducción de resistencias de amortiguamiento en el circuito eléctrico. En el caso de una inductancia, la resistencia habría de disponerse en paralelo con la misma, como se muestra en la siguiente figura, proporcionando un camino alternativo a la intensidad cuando el interruptor se abre. La intensidad en el circuito RL decaerá así con el tiempo.

Se ha de considerar con cuidado el valor de la resistencia. La constante de tiempo RL es:

$$\tau = \frac{L}{R_d}$$

En una primera aproximación, podemos elegir la constante de tiempo en relación con el tiempo de paso Δt , como sigue:

$$R_d = \frac{L}{\alpha \cdot \Delta t}$$

donde α es el factor de amortiguamiento. Un valor de 0.15 para α puede suponer un buen compromiso entre amortiguamiento y exactitud.

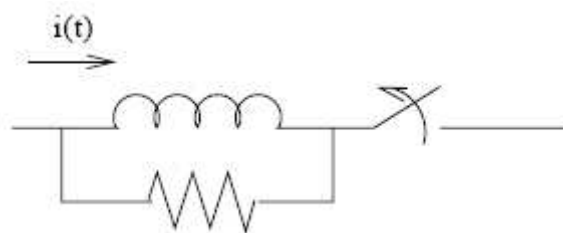


Figura. Inductancia con resistencia de amortiguamiento.

Igualmente puede considerarse para el caso del condensador. En este caso la resistencia se coloca en serie con el condensador, como se muestra en la siguiente figura. Para la elección de la constante de tiempo puede considerarse la siguiente expresión:

$$R_d = \frac{K_s \cdot \Delta t}{2 \cdot C}$$

Se recomienda K_s entre 0.15 y 13.34

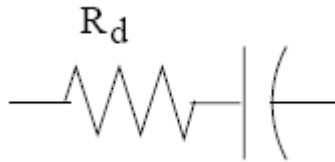


Figura. Condensador con resistencia de amortiguamiento.

A.11.4.5 Interpolación.

El esquema de interpolación detecta el cambio de paso de la intensidad y la tensión cuando se presentan oscilaciones. Entonces considera el tiempo anterior a la misma y realiza una interpolación lineal a la intensidad cero o en la operación del interruptor. Las ecuaciones diferenciales se resuelven en este punto y se comienza de nuevo en el siguiente tiempo de paso.

Netomac fue probablemente el primero de los programas EMTP que incluyeron interpolación.

También se considera interpolación para el uso de modelos de líneas de transmisión y pulsos de disparo en dispositivos electrónicos de potencia.

A.11.4.6 Cambiando el método de integración.

La versión de EMTP de DCG-EPRI (EMTP version 3.0, también conocida como EMTP96), elimina el problema de las oscilaciones numéricas ejecutando dos pasos de tiempo mediante el método de integración 'Backward Euler'.

El usuario invoca el método de integración escribiendo 'CDA' (*Critical Damping Adjustment*) a continuación del BEGIN NEW DATA CASE.